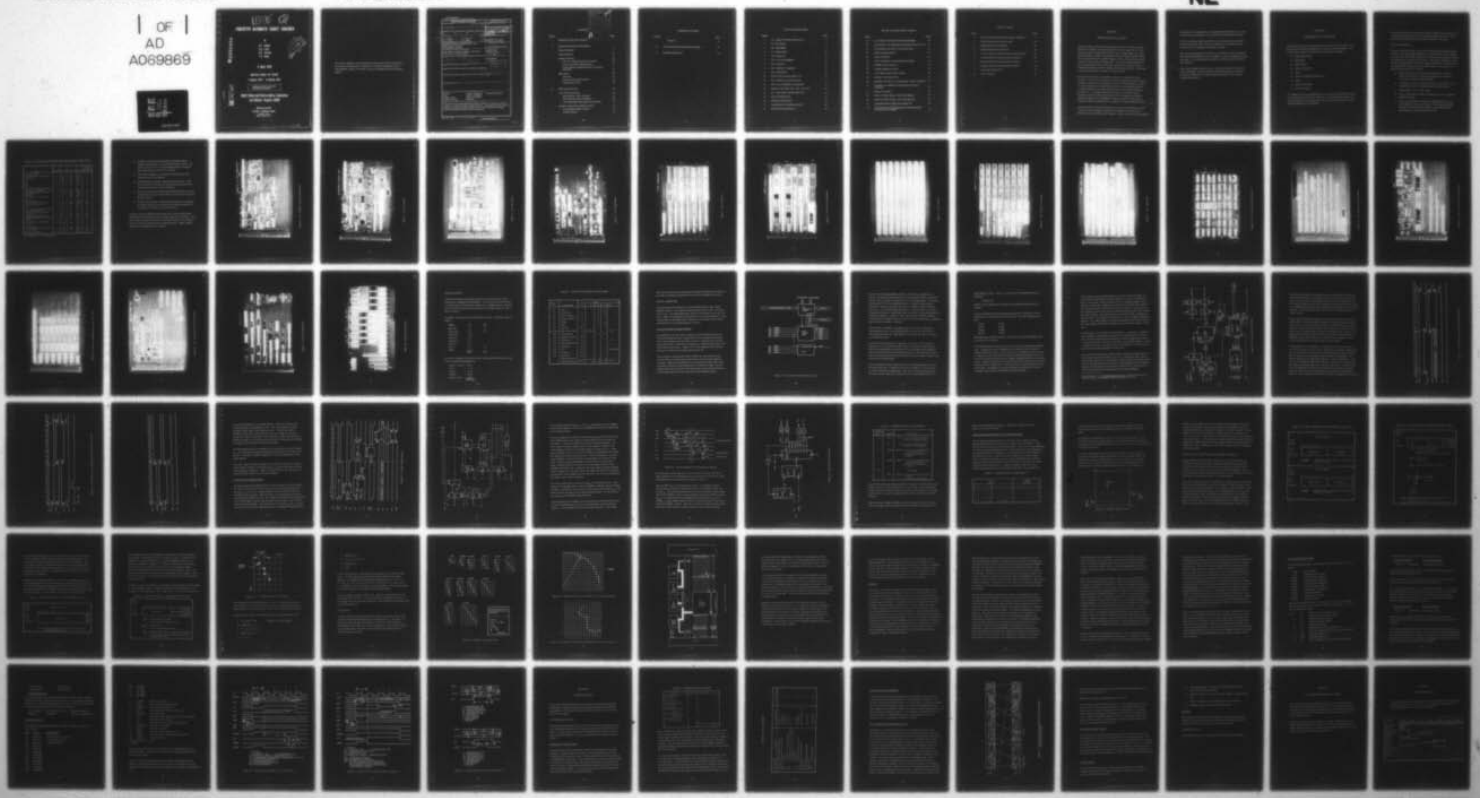
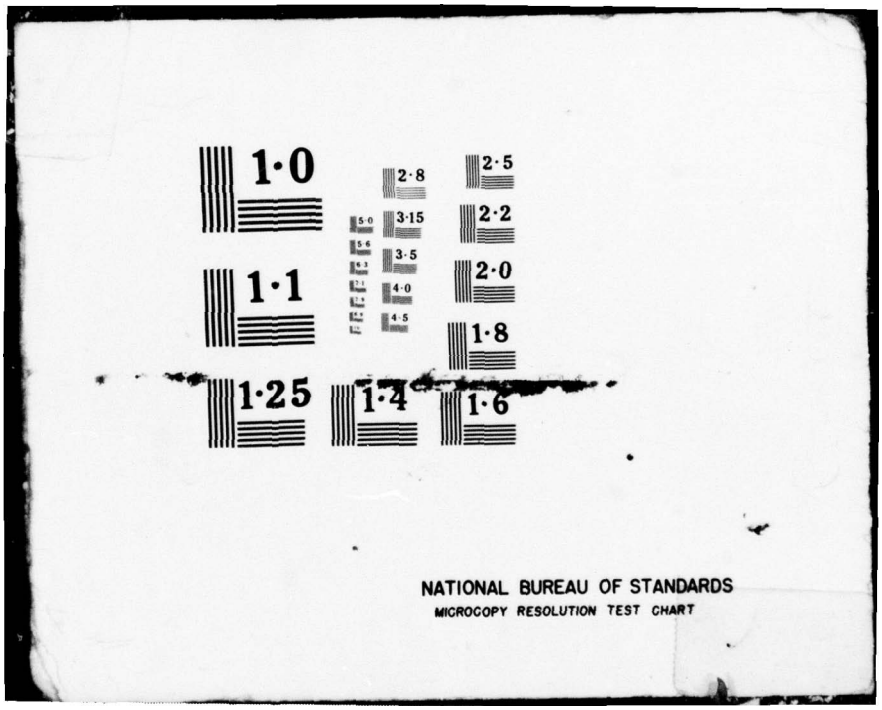
AD-A069 869 HONEYWELL INC MINNEAPOLIS MINN SYSTEMS AND RESEARCH --ETC F/G 17/5
PROTOTYPE AUTOMATIC TARGET SCREENER.(U)
APR 79 D E SOLAND, R C FITCH, D V SERREYN DAAK70-77-C-0248
UNCLASSIFIED 79SRC25 NL

1 OF 1
AD
A069869



END
DATE FILMED
7-79
DDC


1·0
1·1
1·25
2·8
3·15
3·5
4·0
4·5
5·0
5·6
1·4
2·5
2·2
2·0
1·8
1·6
NATIONAL BUREAU OF STANDARDS
MICROCOPY RESOLUTION TEST CHART

LEVEL II

PROTOTYPE AUTOMATIC TARGET SCREENER

by

D.E. Soland
R.C. Fitch
D.V. Serreyn
T.G. Kopet

6 April 1979

Quarterly Report for Period

1 January 1979 - 31 March 1979

Night Vision and Electro-Optics Laboratory
Fort Belvoir, Virginia 22060

Honeywell
SYSTEMS & RESEARCH CENTER
2600 RIDGWAY PARKWAY
MINNEAPOLIS, MINNESOTA 55413

"The views, opinions, and/or findings contained in this report are those of the authors and should not be construed as an official department of the Army position, policy, or decision, unless so designated by other documentations."

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) PROTOTYPE AUTOMATIC TARGET SCREENER	5. TYPE OF REPORT & PERIOD COVERED Quarterly Progress Report 1 January-31 March 1979
	6. PERFORMING ORG. REPORT NUMBER 79SRC25
7. AUTHOR(s) D. E. Soland, D. V. Serreyn, R. C. Fitch, T. G. Kopet	8. CONTRACT OR GRANT NUMBER(s) DAAK70-77-C-0248
9. PERFORMING ORGANIZATION NAME AND ADDRESS Honeywell Systems and Research Center, 2600 Ridgway Parkway, Minneapolis, MN 55413	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 1 E263710DK70/14 010CJ
11. CONTROLLING OFFICE NAME AND ADDRESS Night Vision and Electro-Optics Laboratory, Fort Belvoir, Virginia 22060	12. REPORT DATE April 6, 1979
	13. NUMBER OF PAGES 81
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 1E263710DK70 14	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited.

12 82p. 11 6 Apr 79

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Infrared	Target recognition	Image enhancement
FLIR	Pattern recognition	
Target cueing	Image processing	
Target screening	Real time	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report is the sixth quarterly progress report for contract DAAK70-77-C-0248, Prototype Automatic Target Screener. The objective of the effort is to design an automatic target screener to be used with thermal imaging systems employing common module components.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

402 349

JOB

CONTENTS

CONTENTS (concluded)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (concluded)

LIST OF TABLES

SECTION I

INTRODUCTION AND SUMMARY

This is the sixth quarterly technical progress report for contract number DAAK70-77-C-0248, Prototype Automatic Target Screener (PATS). The first two quarterly reports document the Phase I design study. The third quarterly report provides a description of the final target classifier design for the target data base currently available and the results of the hardware and CPU1 software system design tasks. The fourth and fifth reports and this one describe further the subsystem design details and status of the hardware fabrication, software coding, and hardware checkout. This report covers the period from 1 January to 31 March, 1979.

The program objective is to produce a design for an automatic target screener. The screener will reduce the task loading on the thermal imager operator by detecting and recognizing a limited set of high-priority targets at ranges comparable to or greater than those for an unassisted observer. A second objective is to provide enhancement of the video presentation to the operator. The image enhancement includes: (1) automatic gain/brightness control to relieve the operator of the necessity to continually adjust the display gain and brightness controls and (2) DC restoration to eliminate artifacts resulting from ac coupling of the infrared (IR) detectors.

Image enhancement will also include local area gain and brightness control to enhance local variations of contrast and compress the overall scene dynamic range to match that of the display. This circuitry has been completed,

and examples of its performance on videotaped thermal image data, along with the circuit description, were included in the first quarterly report.

The DC restoration image enhancement circuit eliminates the streaking associated with loss of line-to-line correlation on the displayed image because of the ac coupling of the detector channels.

Some of the high speed parts required in the checkout of CPU1 operations have had extended delivery times; therefore, a contract extension has been necessary. A revised program schedule is included to show the effect of this extension.

[CONT'D]

This report consists of five sections. Section II describes the hardware status and design. Section III presents the software status. Section IV provides the plans for next reporting period and Section V the revised schedule information.

SECTION II

HARDWARE STATUS AND DESIGN

This section describes those design tasks completed and the checkout status of boards whose design description was reported in previous reports. The PATS hardware tasks were broken down into subparts as follows:

- Image Enhancement
- Edge Signal
- Bright Signal
- Interval Generation
- CPU1
- Memory 2 (intensity information)
- CPU2
- Symbol Generation
- Sync and Timing

The design tasks that are reported for this period are the symbol generation and the interface between CPU1 and CPU2. The CPU1/CPU2 interface was described briefly in a previous report but this section will present the final design considerations and solution.

The status table given in the previous report is updated in this section, with a brief discussion of items that have been or remain to be done. A power estimate based upon measured and estimated power is also given.

STATUS OF MODULES

Table 1 presents the status to date of the functional subassemblies defined for PATS. The percentage completed is a rough estimate of our progress with each task. One hundred percent means the task is essentially complete but changes may be made during checkout. Some of the hardware has been checked out individually and partially integrated together. However, during final integration we still may have to make changes. The status of each functional subassembly is as follows:

- Image enhancement--two boards completely built, contrast enhancement board to be checked out shortly, DC restore to be checked out during integration phase
- Edge Signal--one board completely built and checked out; a few components need to be changed because of board spacing
- Bright Signal--same as edge signal
- Interval Generation--design not completed yet, to be worked on during next period
- CPU1--six full boards completed, including sequencer, ALU with multiplier, Memory 1, Memory 1 constants, DMA/FIFO interface, and CPU1/CPU2 interface; some required parts still missing but initial checkout has begun

TABLE 1. STATUS OF PATS HARDWARE (PERCENTAGE COMPLETED)

Subpart	Boards	Design	Schematics	Build	Checkout Excluding Integration
Image Enhancement					
Adaptive Contrast Enhancement	1	100	100	100	0
DC Restore	1	100	100	100	0
Edge	1	100	100	100	95
Bright	1	100	100	100	95
Interval	1	65	0	0	0
CPU1 (Digital Processing Subsystem)					
Processor Inc Multiplier	1	100	100	100	0
Microprogram Memory Sequencer	1	100	100	100	50
FIFO/DMA I/F	1	90	95	95	0
Memory 1	2	95	100	100	0
CPU1/CPU2 I/F (inc in CPU2)	1	NA	NA	NA	90
Memory 2					
A/D, Summation	1	100	100	100	100
Memory Control and Refresh	1	100	100	100	100
Memory 512 x 512 x 2	4	100	100	100	100
CPU2					
CPU with 16K Memory (KD 11-HC)	2	NA	NA	NA	100
PROM Board (MRV11-AA)	1	NA	NA	0	0
Serial Port (DLV11) inc Printer/ Keyboard	1	NA	NA	NA	100
Refresh/Bootstrap (REV11-C)	1	NA	NA	NA	100
Floppy Controller (RXV11-BA) inc Floppies	1	NA	NA	NA	100
Symbol Generator (one included in CPU2)	2	100	100	10	0
Sync and Timing					
Sync Separator and Video Switches	1	100	100	100	100
Sync Generation	1	100	90	100	100
Writable Control Store	1	100	0	100	100

NA-Not applicable to PATS Design Tasks

- Memory 2--total of six boards built and initial checkout completed on all boards; this includes memory control, A/D boards, and 512 x 512 x 2 memory planes (four); presently only implementing a 512 x 256 x 2 memory
- CPU2--this computer was purchased; all parts have been assembled and are functioning
- Symbol Generator--design complete for both boards; build for one board has started; one board will fit into main PATS chassis and the other in CPU2 mainframe
- Sync and Timing--both boards built and checked out; work well with 525-line system; PROM will have to be blown for use with 875-line system
- Writable Control Store--additional board which is not included in PATS hardware but necessary for microcode checkout; all checked out and being used in CPU1 checkout

In order to provide additional information on the status of the boards, pictures of those already built are included as part of this section. These boards will eventually be put into the ATR-type chassis. These figures are in the order in which they were mentioned above. Figure captions indicate the assigned slot in the chassis.

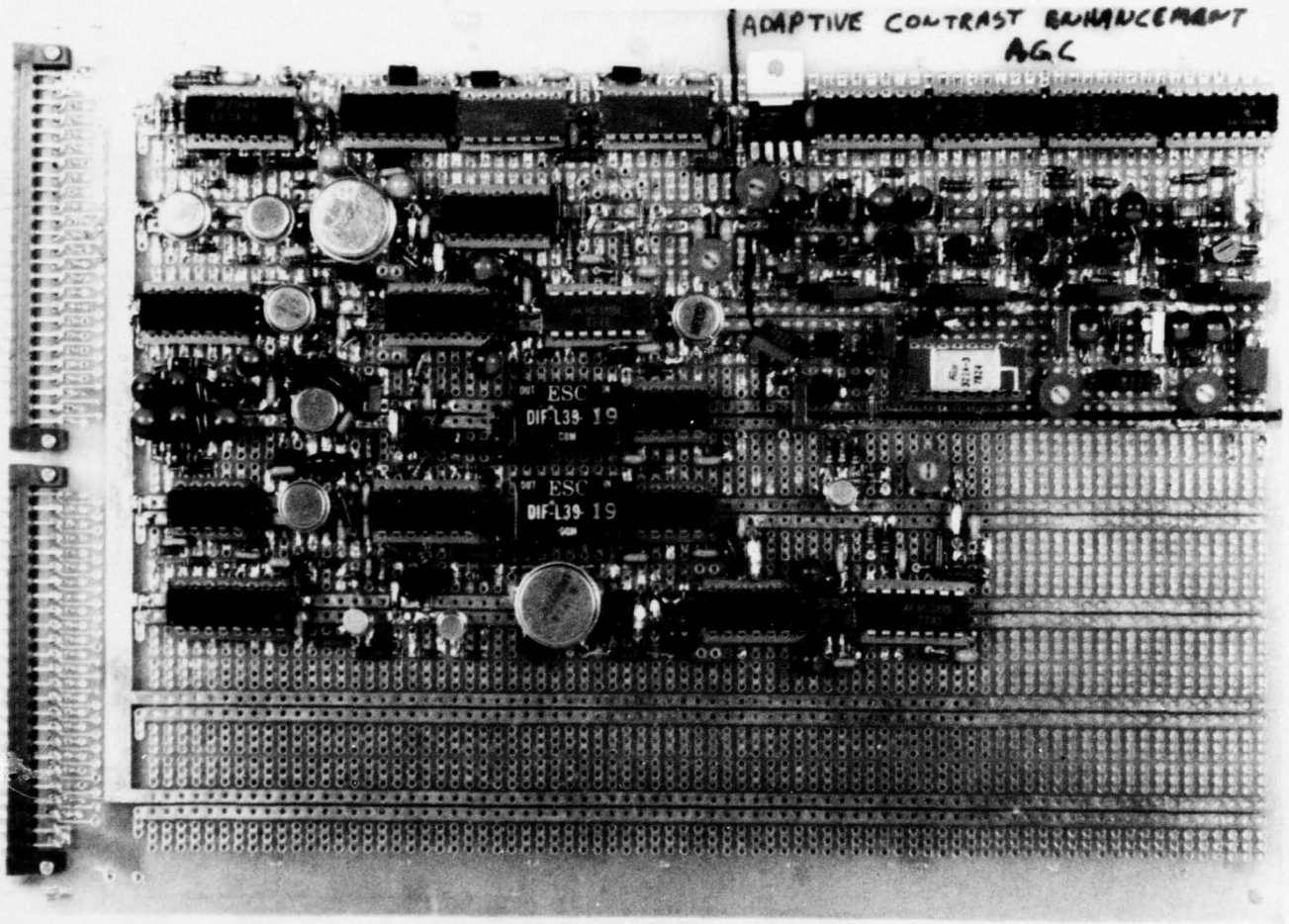


Figure 1. A1: Adaptive Contrast Enhancement

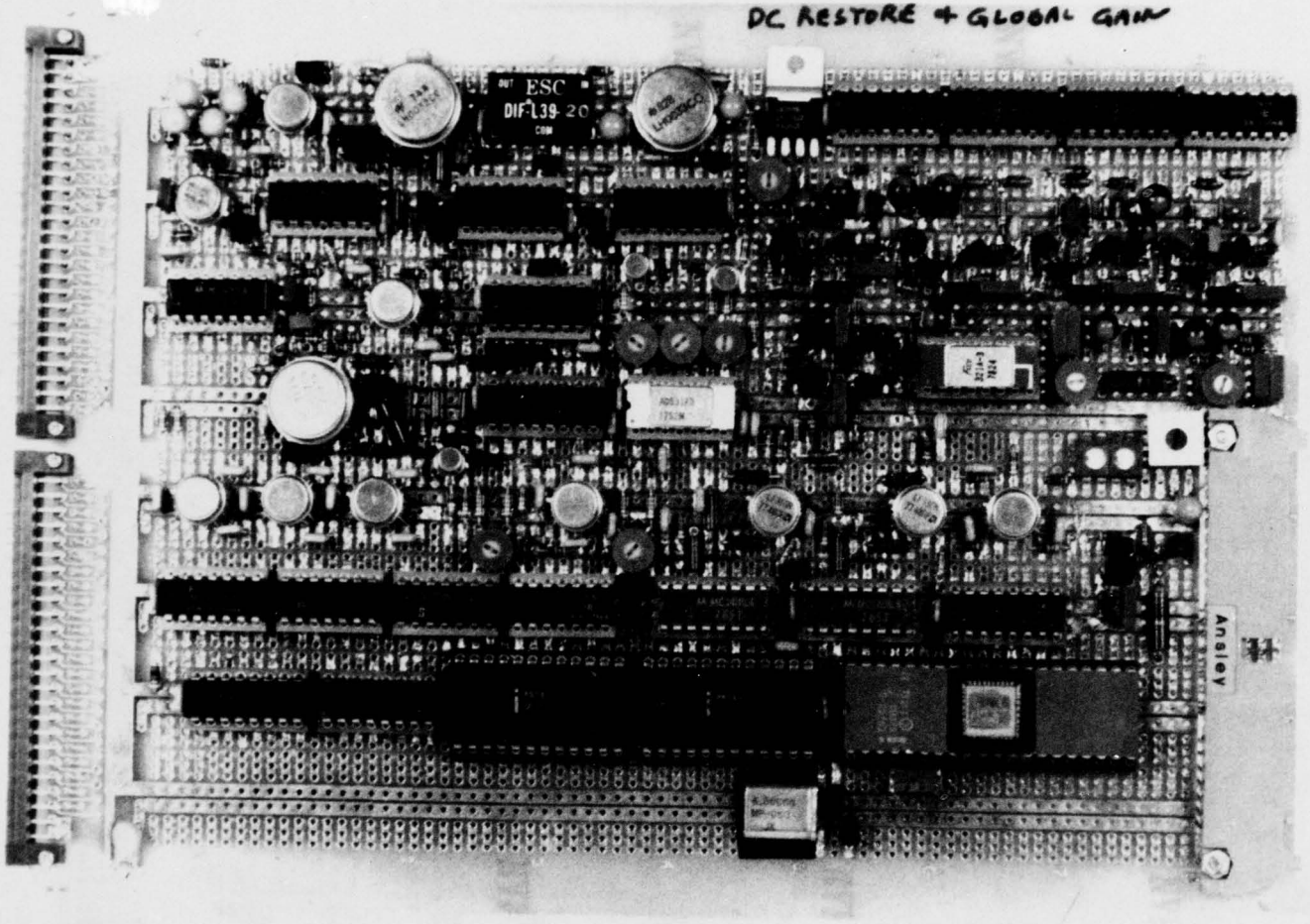


Figure 2. A2: DC Restore

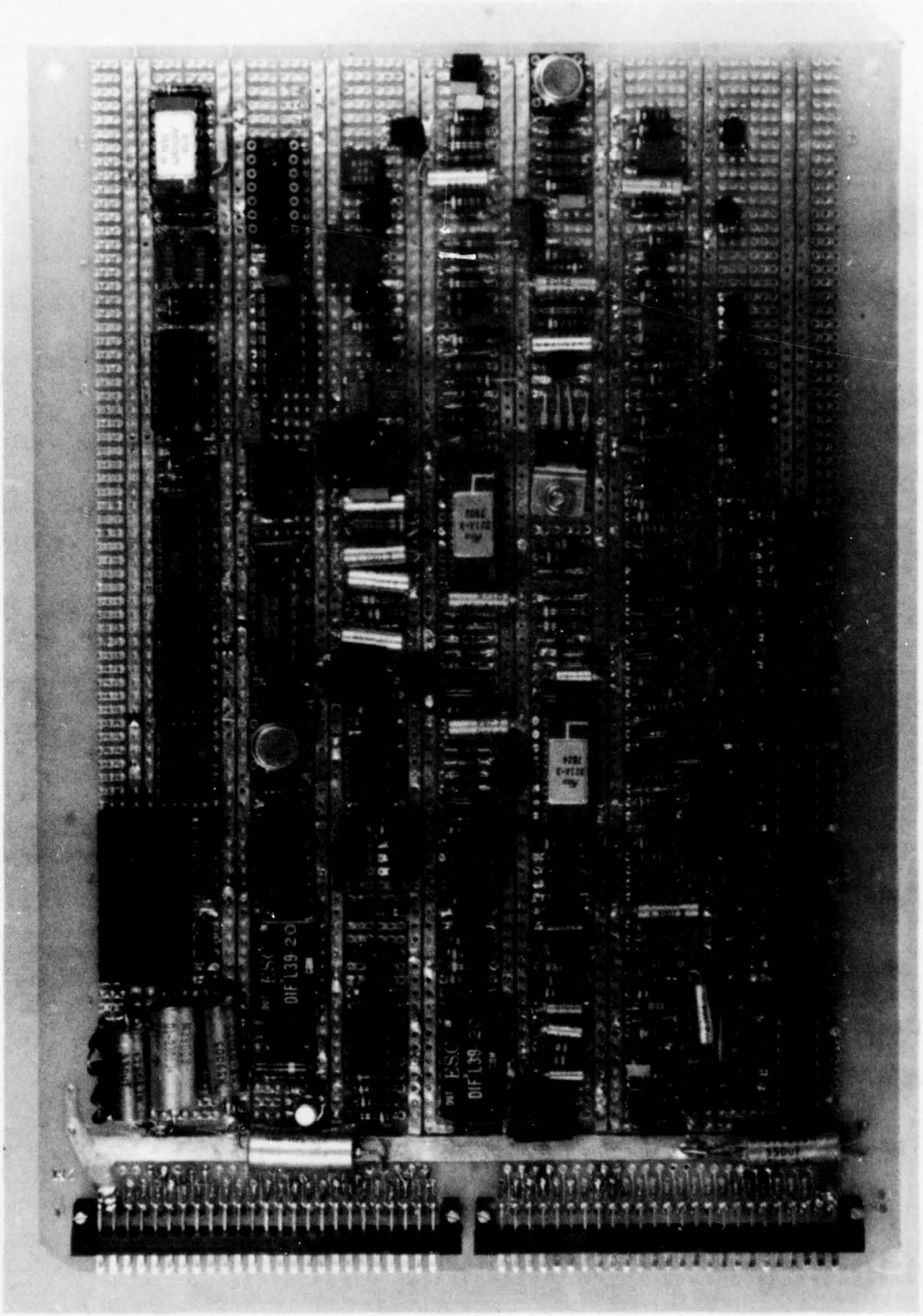


Figure 3. A6: Edge Signal

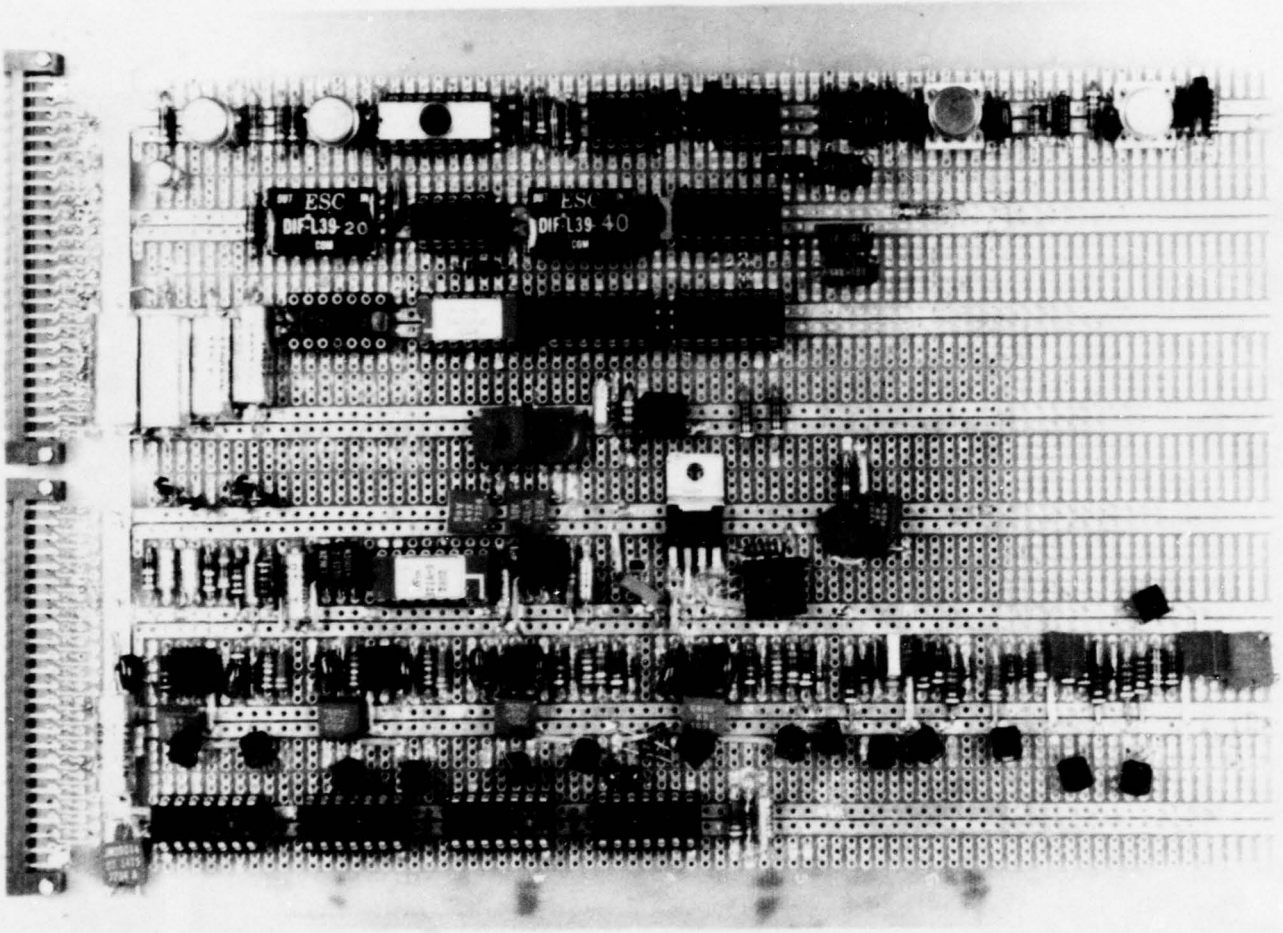


Figure 4. A5: Bright Signal

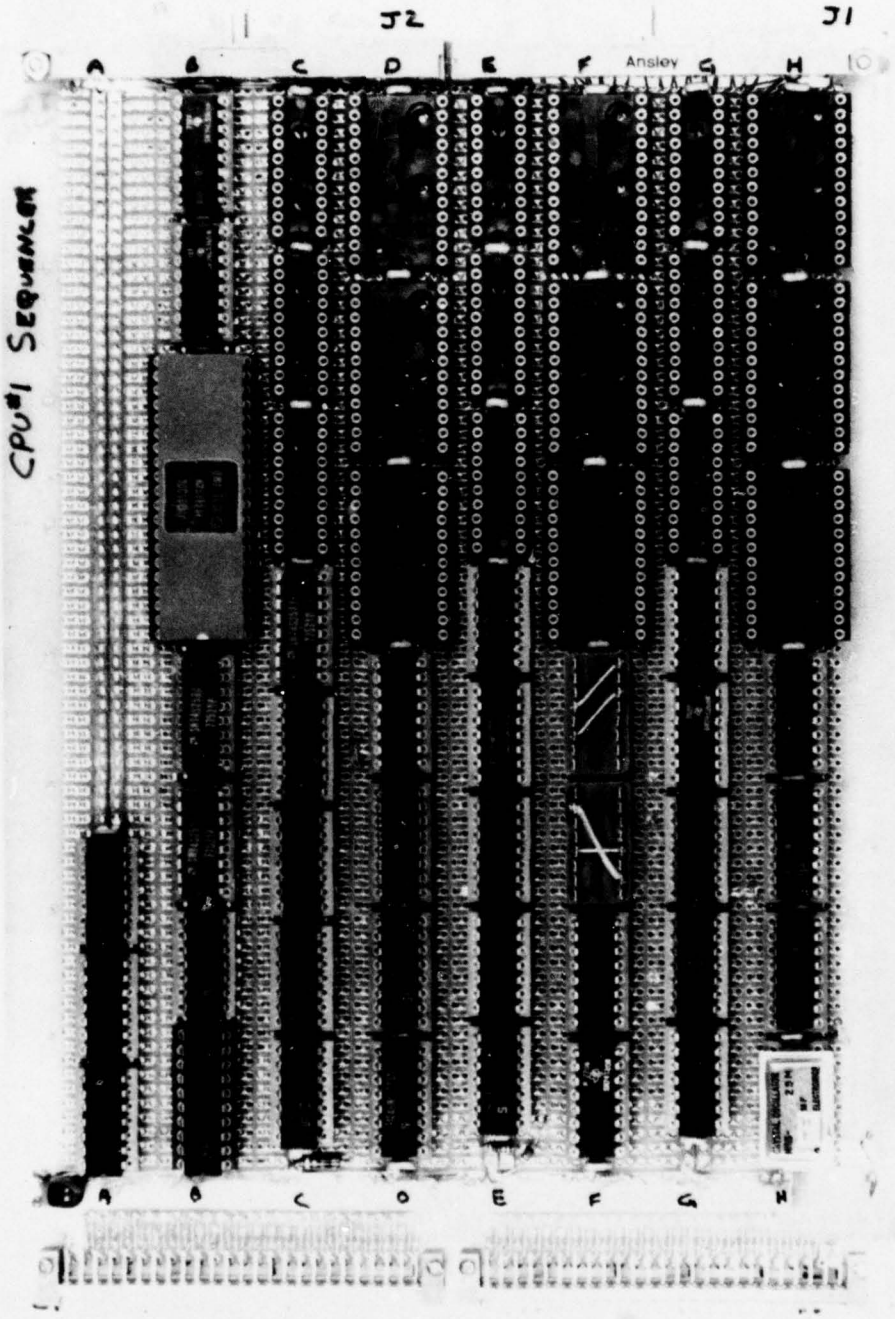


Figure 5. A32: Sequencer

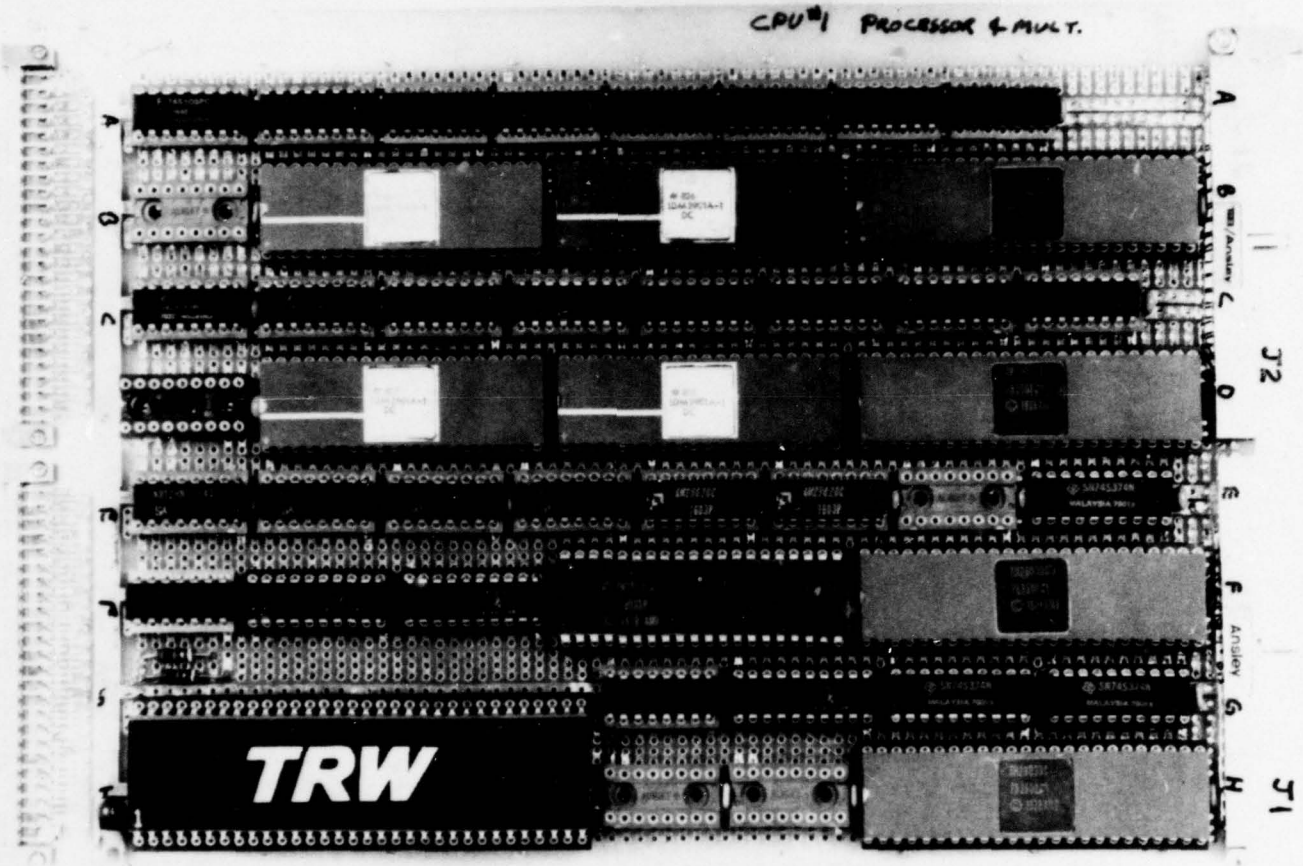


Figure 6. A34: ALU and Multiplier

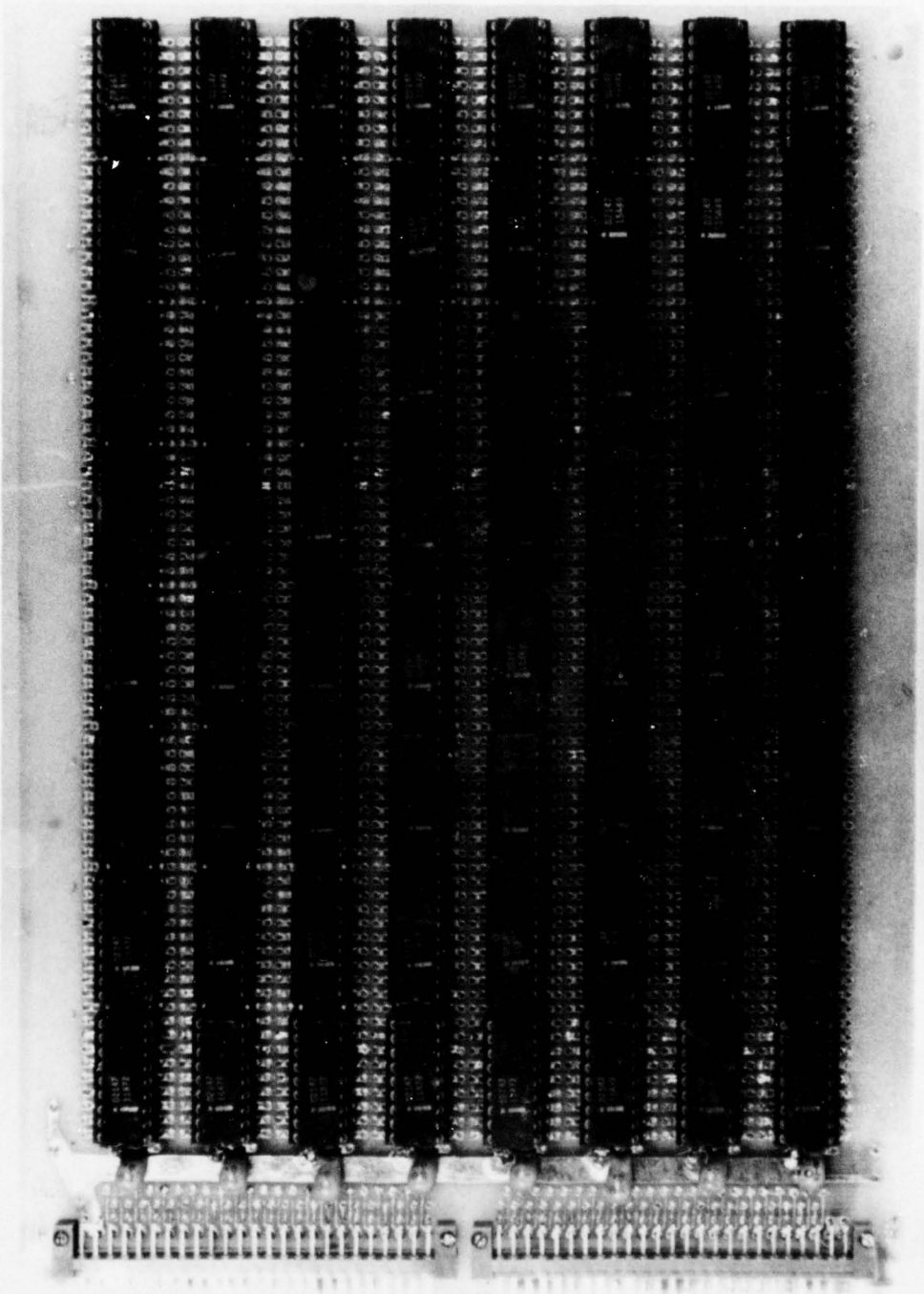


Figure 7. A37: Memory 1

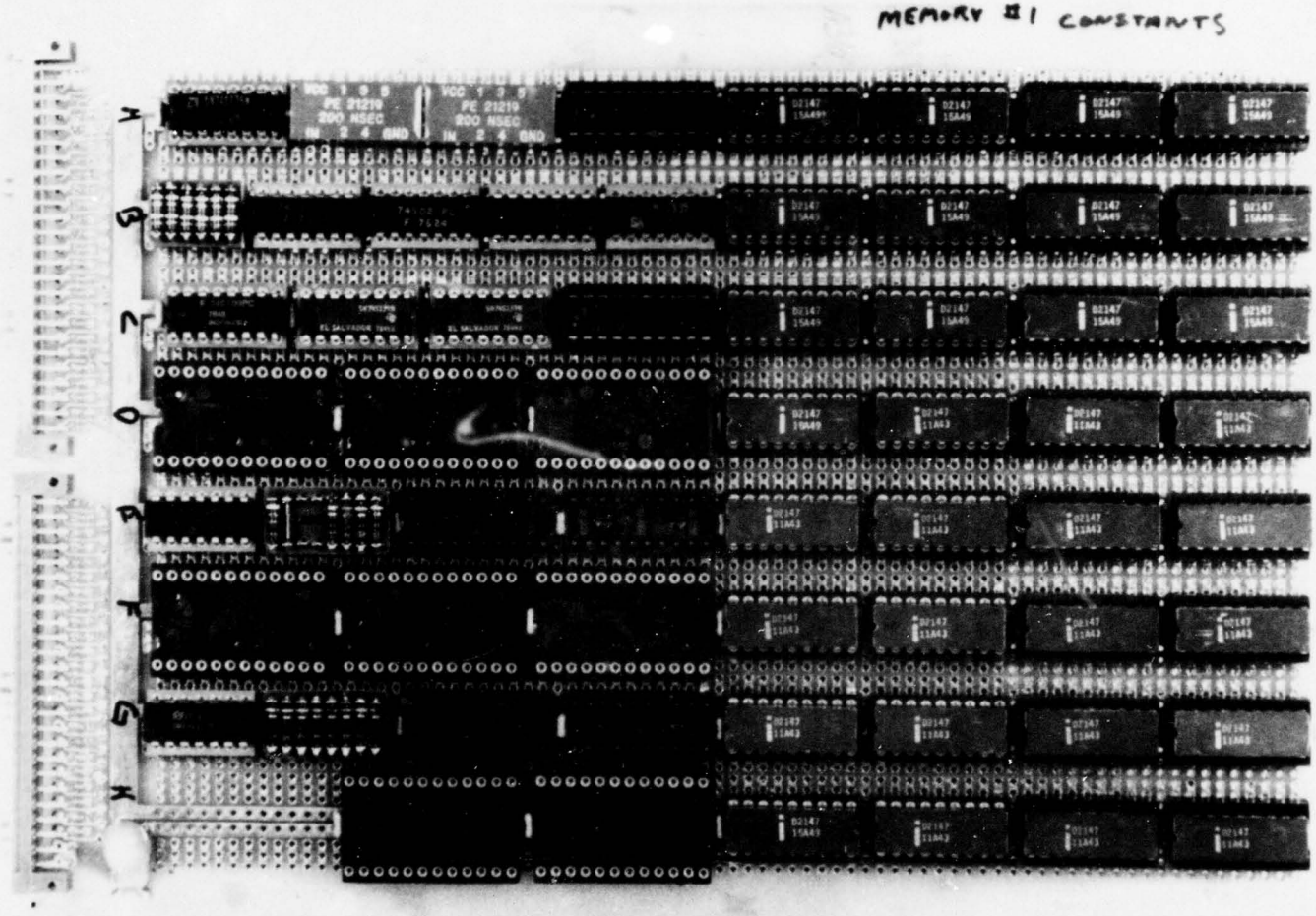


Figure 8. A36: Memory 1 Constants

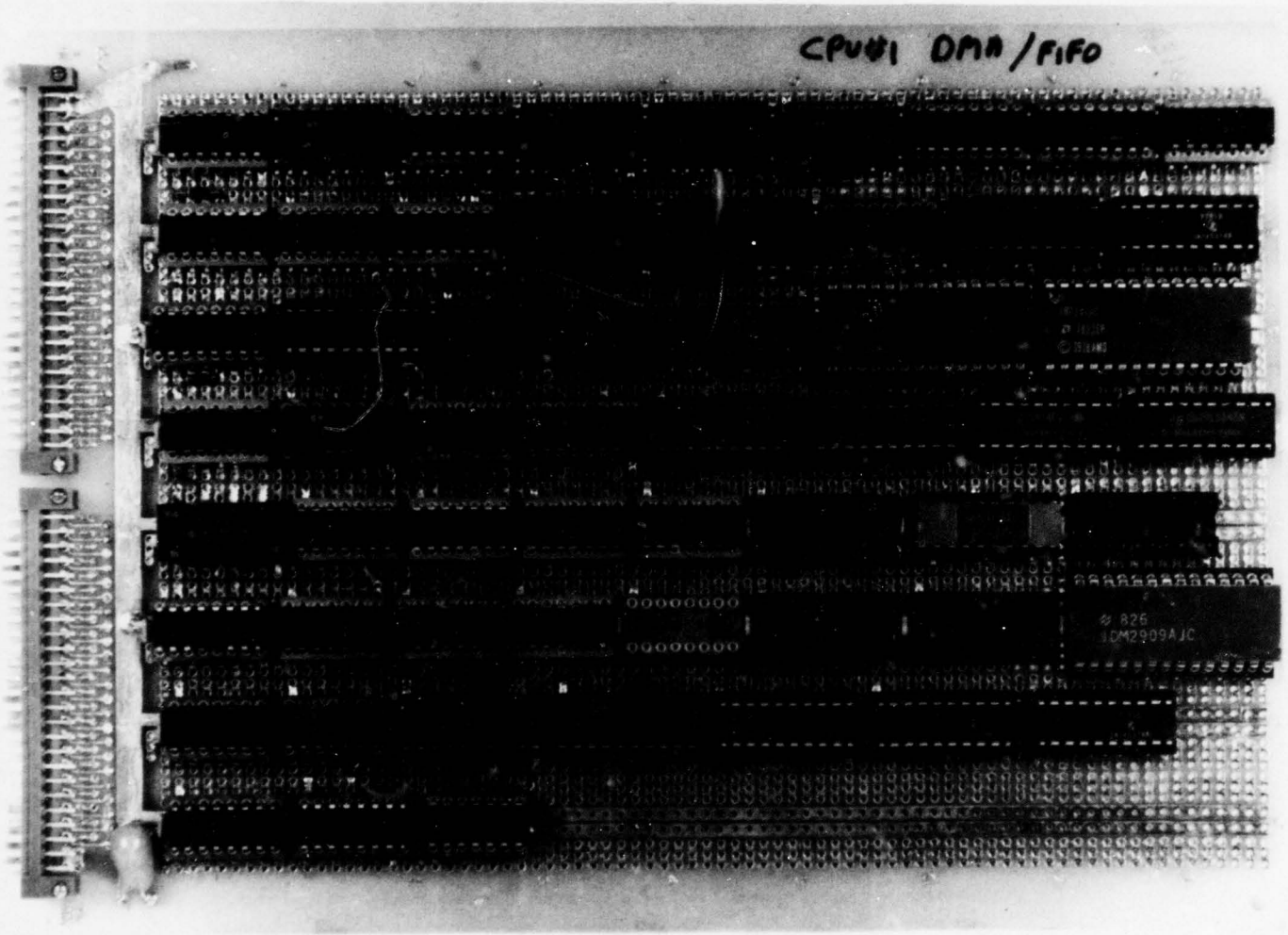


Figure 9. A31: DMA/FIFO

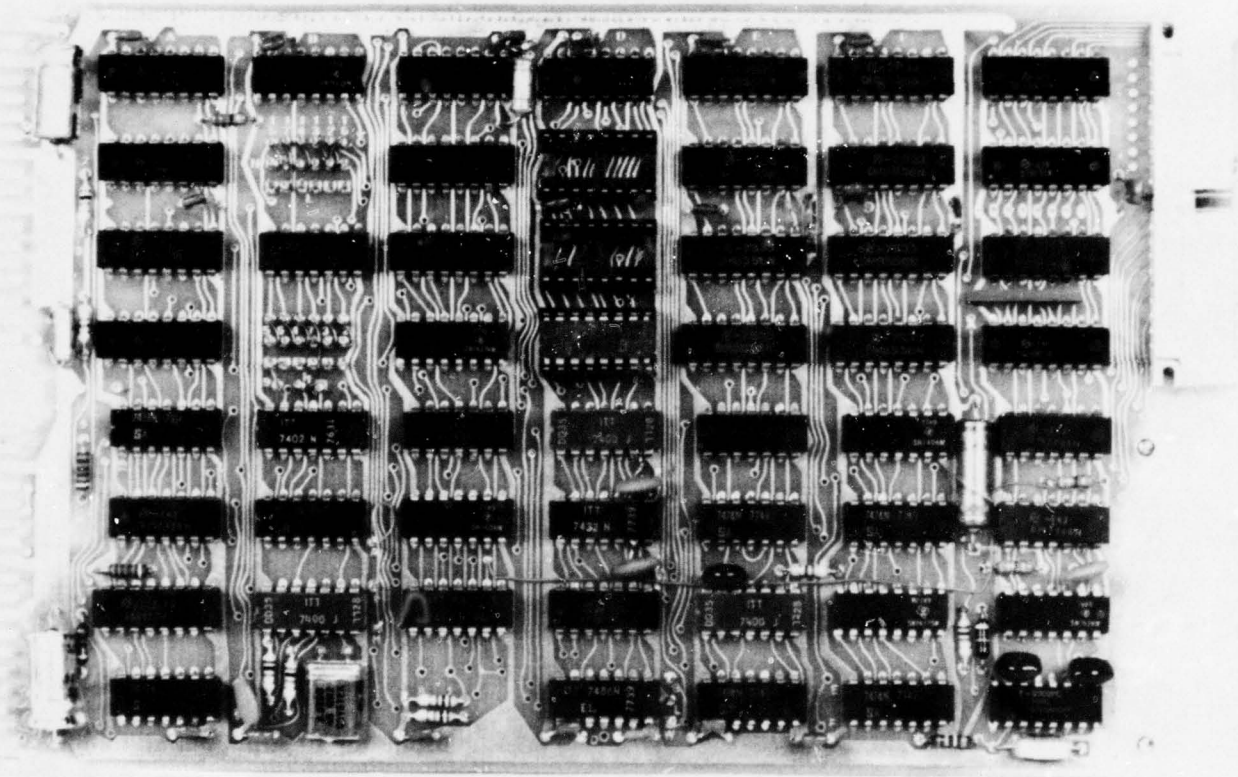


Figure 10. CPU1/CPU2 Interface (DMA-L11)

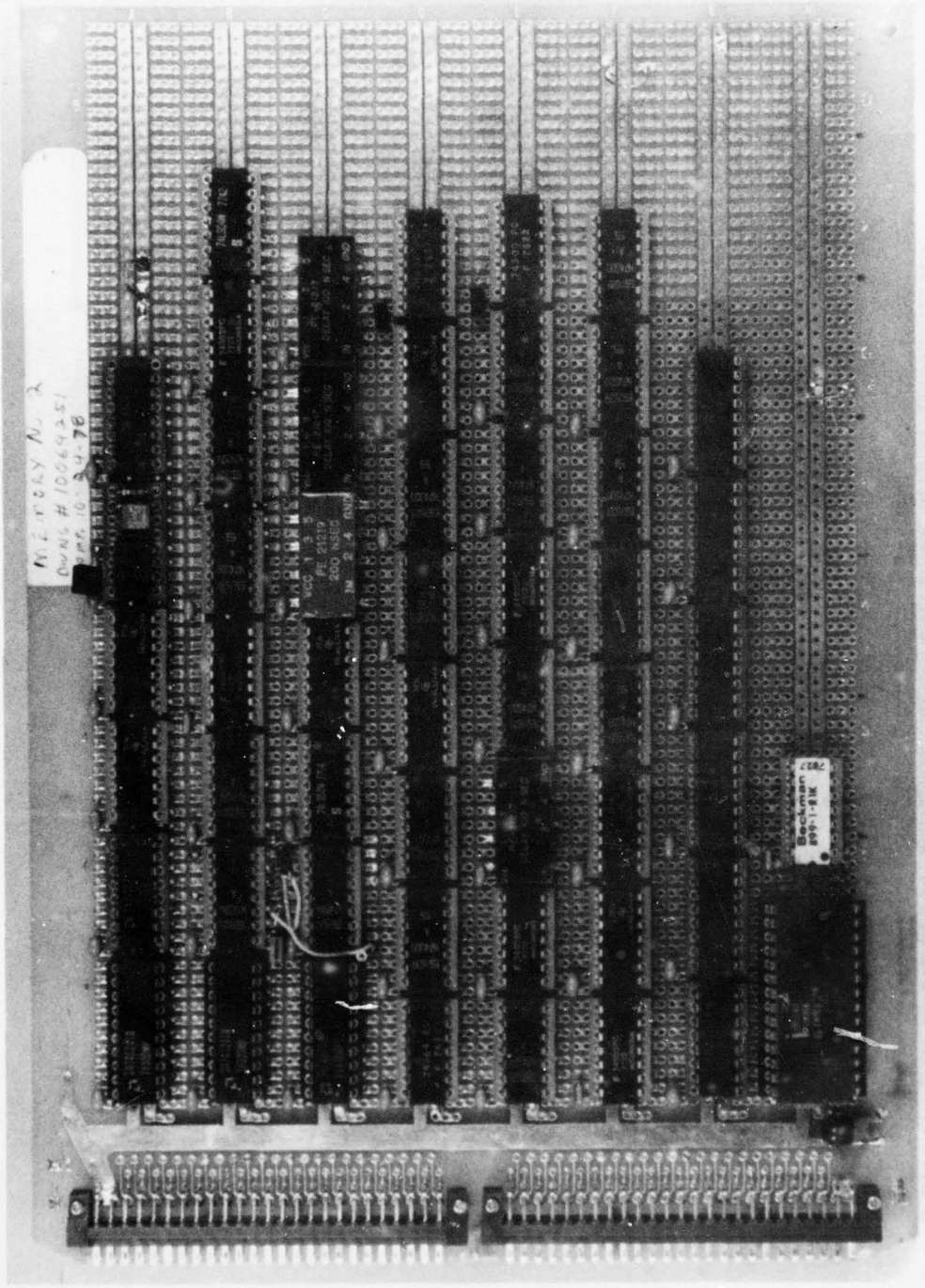


Figure 11. A27: Memory and A/D Control

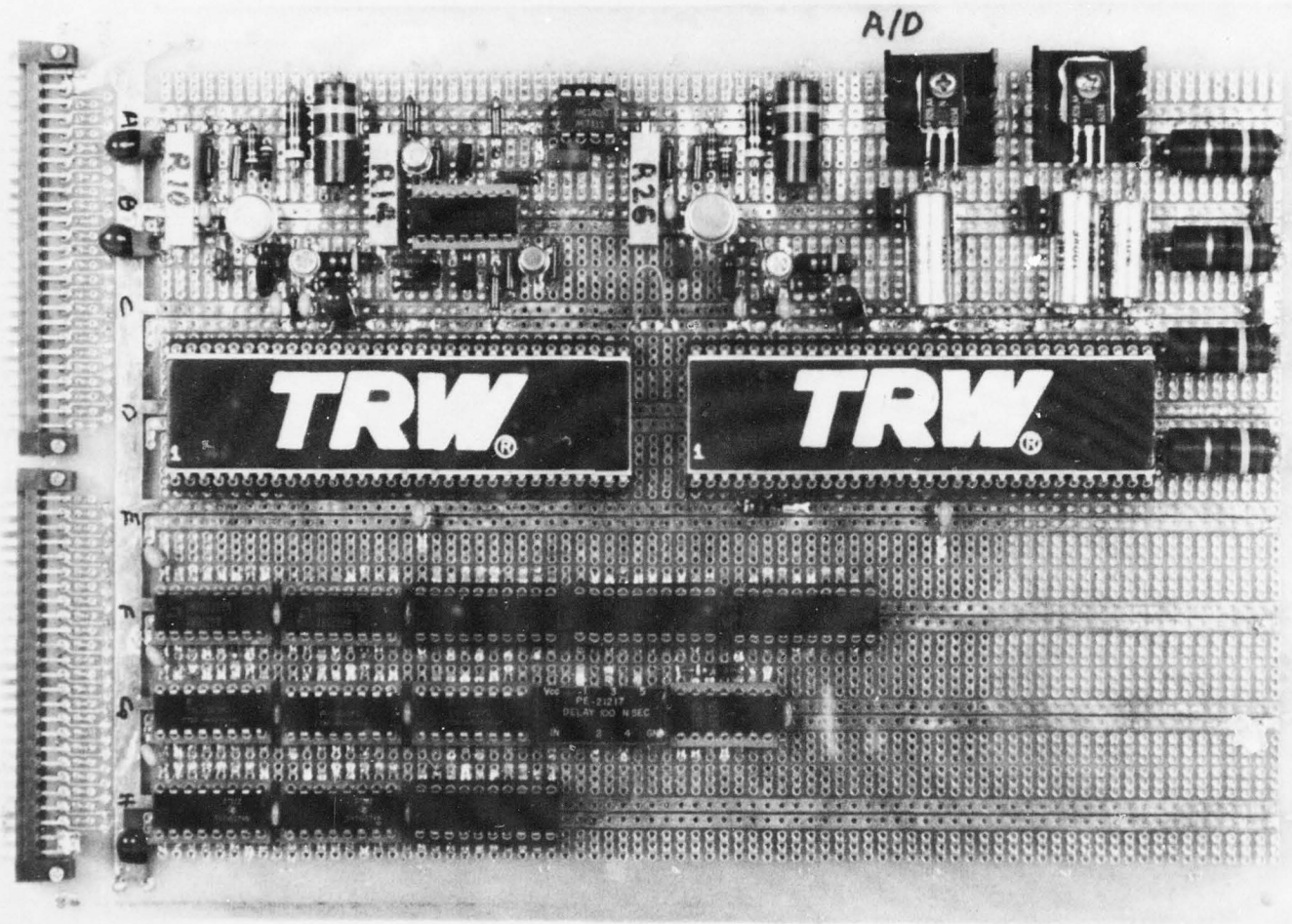


Figure 12. A26: A/D, Summation, Background

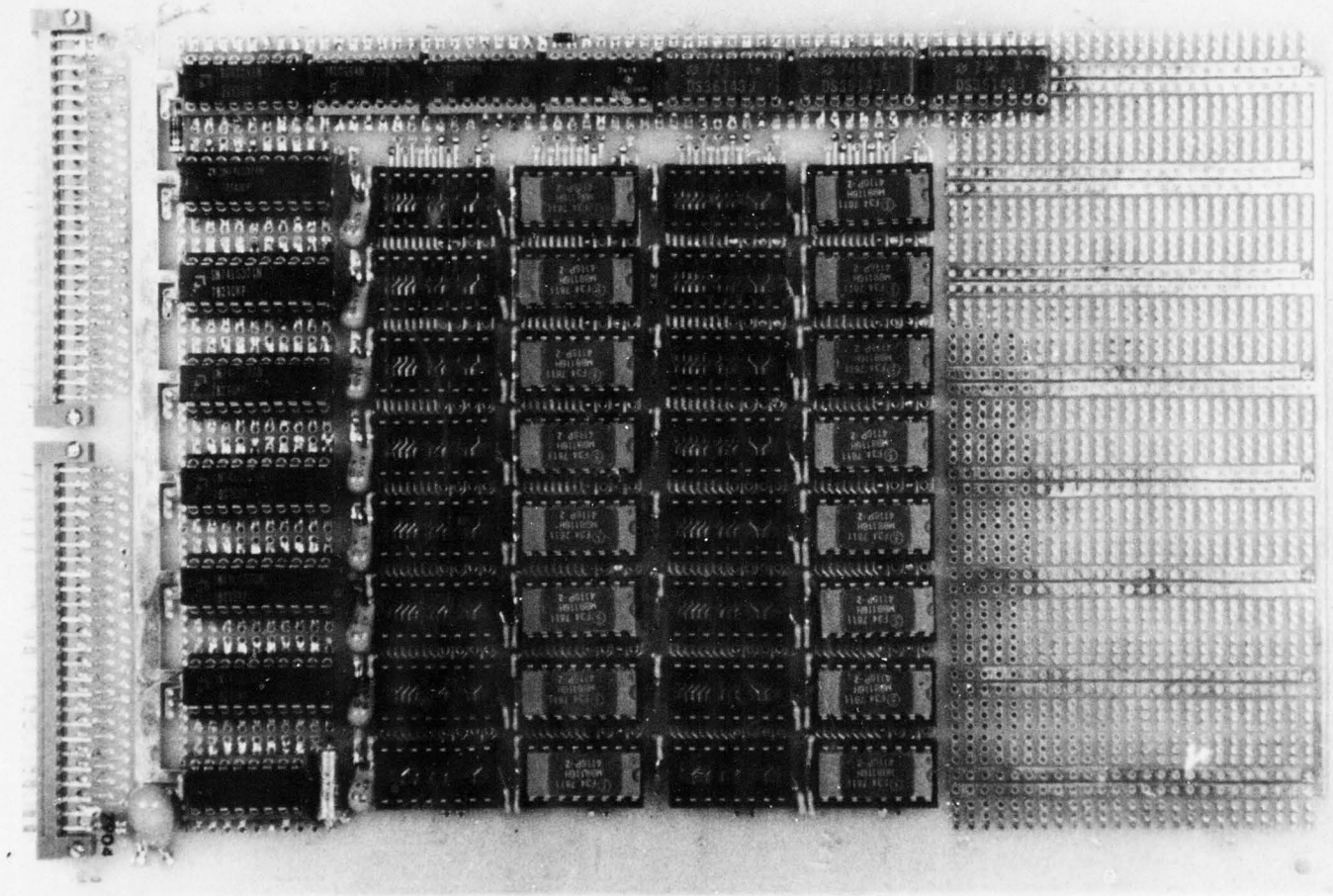


Figure 13. Memory 2 Bit Plane (A21, A22, A23, A24)

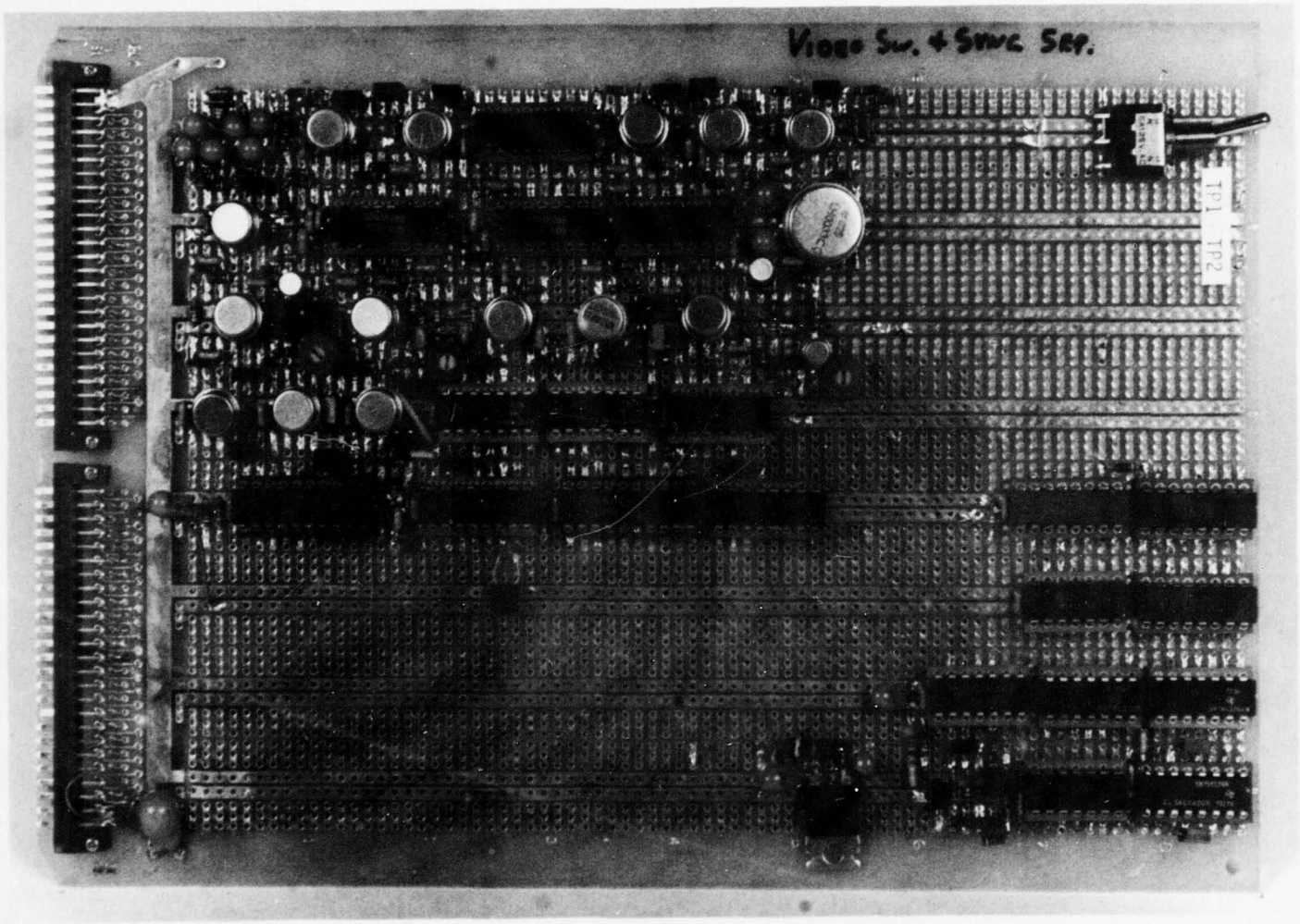


Figure 14. A3: Video Switch and Sync Separator

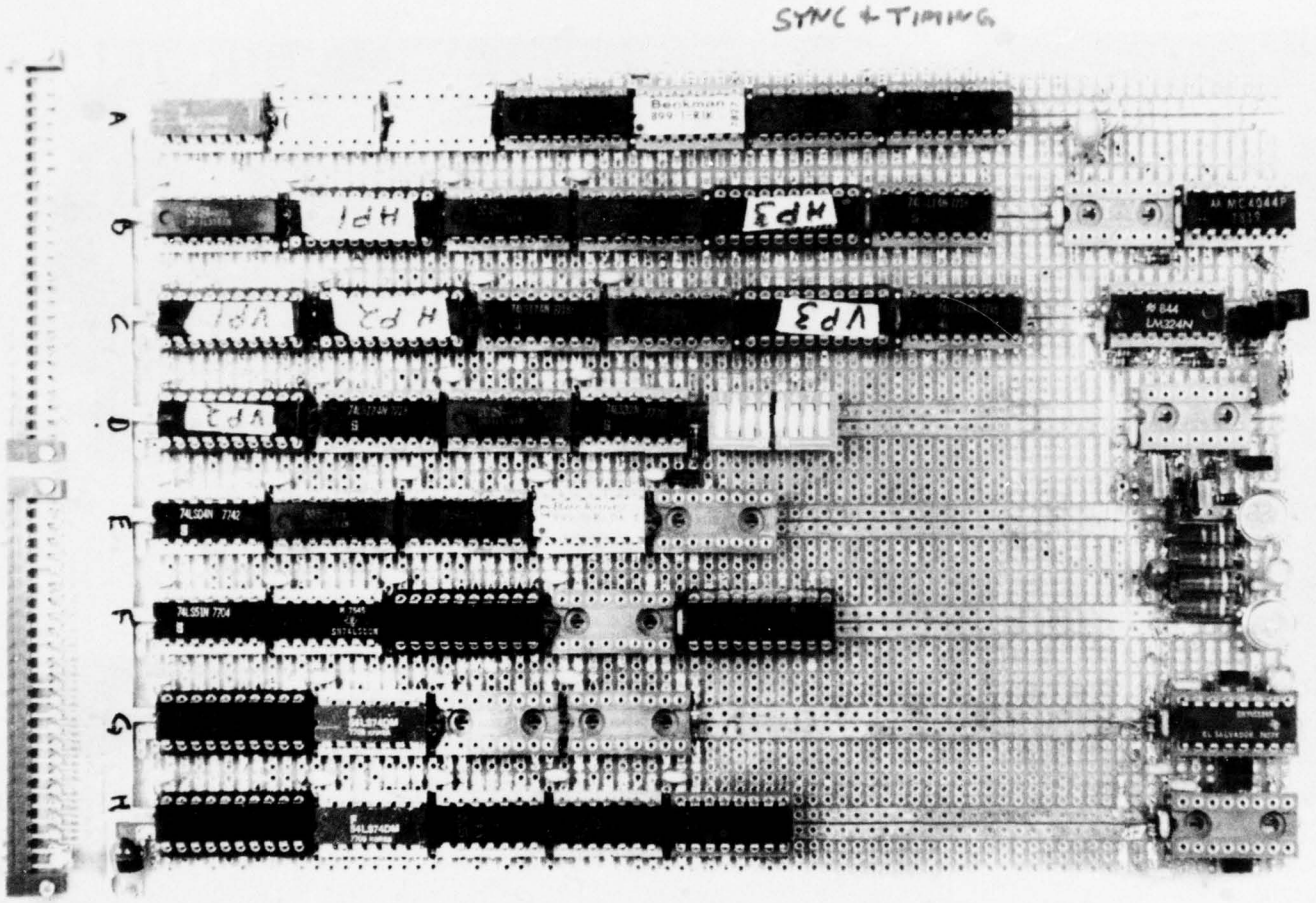


Figure 15. Sync and Timing (A4)

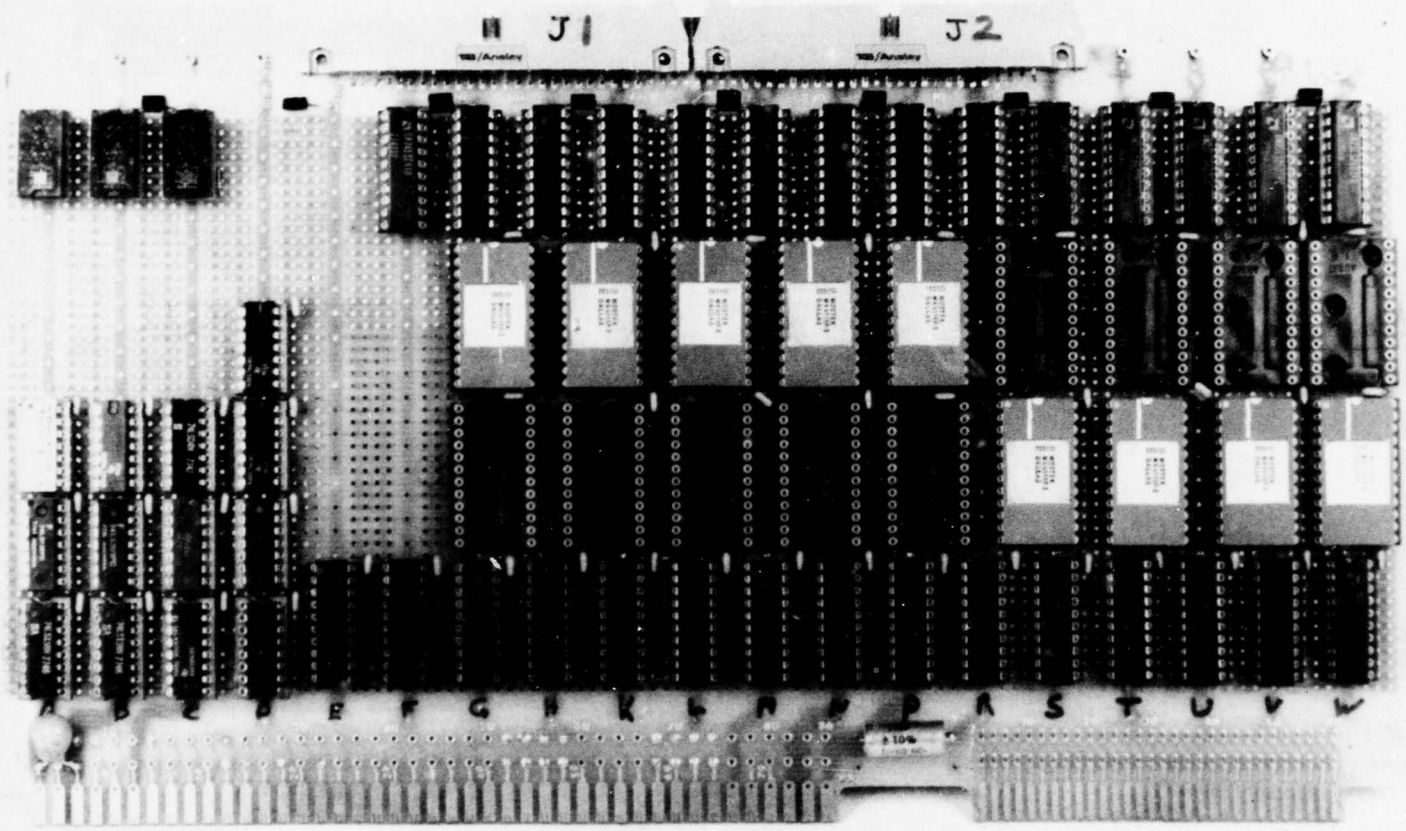


Figure 16. Writable Control Store

POWER ESTIMATE

This power estimate is based upon measurements of the current from various power supplies during checkout. Not all parts were in the boards at checkout time so the measured values may be slightly biased. However, the power for these parts has been estimated and added to give the values listed in Table 2.

In addition, CPU2 power must be taken into account. This breaks down as follows:

Board	+5	+12
KD11-HA	1.6	.25
MSV11-DC	1.8	.36
MRV11-AA	3.45	
DMA-L11	2.0	
Symbol I/F	2.0	
REV11-C	1.5	
DLV 11	1.6	.22
RXV11	1.5	
	15.45	.83

The power calculated for purely dc loads (including the CPU2 boards which are not required for operation) is:

+5(22.6 + 15.45)	=	190.3
-5(.1)	=	.5
+15(1.6)	=	24.0
-15(2.0)	=	30.0
+12(2.5 + .83)	=	40.0
		≈285 watts

TABLE 2. POWER ESTIMATION (current in amps)

Board #	Description	Volts				
		+5	-5	+15	-15	+12
A1	ACE	.1		.3	.2	
A2	DC Restore	.2		.3	.2	
A3	Video Switches	.2		.2	.1	
A4	System Timing	1.3		.1		
A5	Bright	.1		.3	.2	
A6	Edge	.1		.3	.2	
A7	Interval	2.0*				
A21-A24	Mem 2 Bit Plane	.1 x 4	.025 x 4			.5 x 4
A26	A/D Summer	.8		.1	1.1	
A27	Memory Control	1.4				
A28	Symbol Gen	1.0*				.5*
A31	Input DMA/FIFO	3.0*				
A32	Sequencer	3.0*				
A34	ALU	3.0*				*estimated
A36	Memory 1 C	3.0*				
A37	Memory 1	3.0*				
	Total in Chassis					
	Total in Chassis	22.6	.1	.16	2.0	2.5

This does not take into account the power supply inefficiencies or safety factor which we will have to include in our final power supply specification.

SYMBOL GENERATOR

The symbol generator function in PATS provides the user with a video graphics overlay. It is capable of operating with either a 525- or 875-line raster scan. The symbol generator consists of an interface to the LSI 11/2 bus, a 256K x 1-bit graphics overlay memory and a vector generator. Each of these parts is described below, followed by a statement of word format requirements for the symbol generator software.

LSI 11/2--Symbol Generator Interface

The interface to the DEC LSI 11/2 bus is provided almost entirely by six LSI chips developed by DEC specifically for general purpose interfacing. These chips minimize the chip count required to implement bus circuitry. The chips used are the DC003 (interrupt logic), DC004 (protocol logic), and DC005 (transceiver logic). Figure 17 is a block diagram of the interface between the computer bus and the symbol generator.

The transceiver provides data lines to reflect the state of the bus when receive (REC) is asserted and to drive the bus when transmit (XMIT) is asserted. Address and interrupt vector information for interrupt request and device selection is also provided by the transceiver chips. When the address lines match the state of the associated bus line this will allow the protocol logic chip to look for the proper synchronization signals.

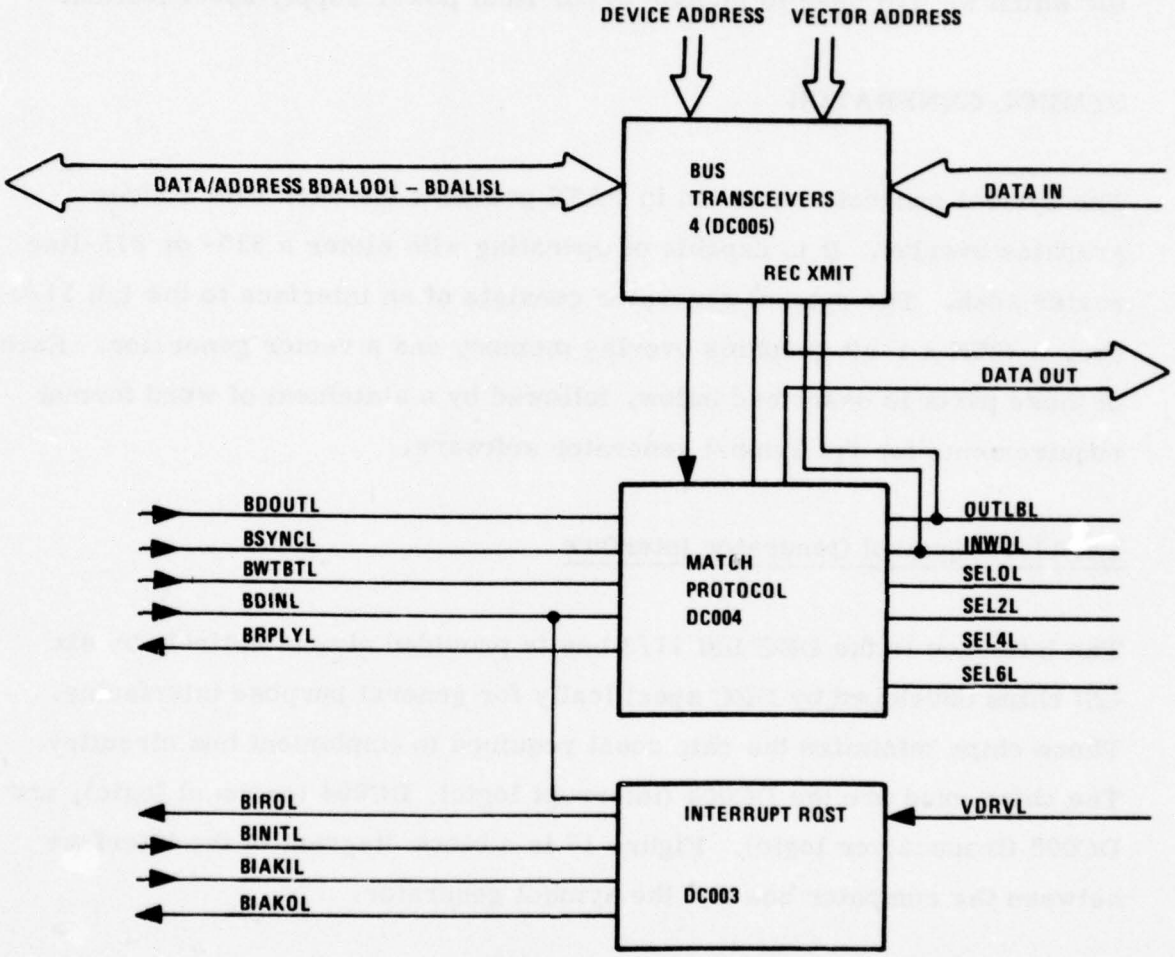


Figure 17. Bus Interface for Symbol Generator

Once the control signals BDINL, BSYNCL, BWTBTL and BDOUTL are present, the protocol logic generates the control signals INWDL, OUTLBL, SELOL, SEL2L, SEL4L, and SEL6L for the symbol generator. To transfer data from the 11/2 to the symbol generator, a BDOUTL, along with the appropriate address on the data bus to give the proper SELXL command, must occur. The protocol logic functions as a register selector to provide the signals necessary to control data flow into and out of the symbol generator register (X-address, Y-address, data, and function). Each SEL output is used to select one register from or to which the data is transferred, depending upon the status of INWDL or OUTBL.

Either BDINL or BDOUTL, depending upon the bus cycle, will initiate a delayed signal back to the bus indicating that the data has been received. This signal-generated bus reply (BRPLYL) will allow the LSI 11/2 to go on to the next instruction.

The interrupt logic performs an interrupt transaction. For the symbol generators, the vertical drive (VDRV) signal provides the main interrupt. When the request (RQST) is asserted, the interrupt logic asserts BIRQL to the bus which initiates the bus handshake operation. This operation terminates with the generators of the vector address by the transceiver logic under control of the interrupt logic.

Data is typically read from and written into the symbol generator by use of the MOV instruction. Usually data is stored in Register 1 in the CPU for a particular symbol to be generated. Since this is a memory-mapped I/O, the output register address is the address of the symbol generator and is

called Register 2 (R2). Hence, to write data to the symbol generator the mnemonic

$$MOV\ R_1,\ R_2$$

applies and the appropriate bus signals are automatically generated and decoded.

The device address for the symbol generator will be 177000 with a vector address for the vertical drive interrupt at 240. The select addresses are then

SEL0L	177000
SEL2L	177002
SEL4L	177004
SEL6L	177006

Depending upon the MOV command, these will be gated as being either read or write select commands.

The symbol generator memory stores 512 x 512 words of single-bit graphics data. Graphics bits can be randomly read or written by CPU2 using the symbol generator interface. The memory can be randomly accessed at about a 1 MHz-rate at the same time the memory is being sequentially read out to the video monitor by interleaving the random and sequential (or video) memory cycles. During vertical and horizontal retrace times, random access can be made at close to a 2 MHz-rate. Low power 16K x 1 dynamic RAMs are used for memory. Normally these memories require a refresh cycle, but the continuous video readout accomplishes this automatically.

A block diagram of the symbol generator memory is shown in Figure 18. The memory is multiplexed to make it appear to be a 512 x 512 single-bit plane. The memory, which uses sixteen 16K x 1 dynamic RAMS, is very similar in architecture to Memory 2, discussed in previous PATS reports.* The addressing for the memory comes from two sources, the sequential or video address counter (VAC) and the random address from the CPU2 symbol generator interface. Selection is made by the address MUX, depending on the cycle type. The address to the memory is a seven-bit split multiplexed address. The random address is already multiplexed on the CPU2 interface. The video address MUX (VAM) multiplexes the 14-bit video address counter (VAC).

The data inputs to the memory are tied together. A single bit is written into the memory by enabling the write line on the selected 16K x 1 memory chip. The data outputs from the memory are latched during video reads in a 16-bit latch and then loaded into a parallel-to-serial shift register. During a random read cycle, the memory data is run through a 16-to-1 line MUX and latched, thus allowing single-bit operations.

The memory timing and MUX control (MTMC) generates timing signals to read and write the memory (RAS, CAS, WE); controls the video address MUX (ENVM, ENVL), the video address counting (CLKX), the address MUX (VCYC), the address MUX on the CPU2 symbol generator interface (ENRL, ENRM), and the memory data outlatches (CVDT, CRDT); and generates a "cycle done" signal to the symbol generator interface (RDONE).

---

*Duane Soland, et al., PATS Quarterly Report #3, Honeywell Systems and Research Center, Minneapolis, Minnesota, September 1978.

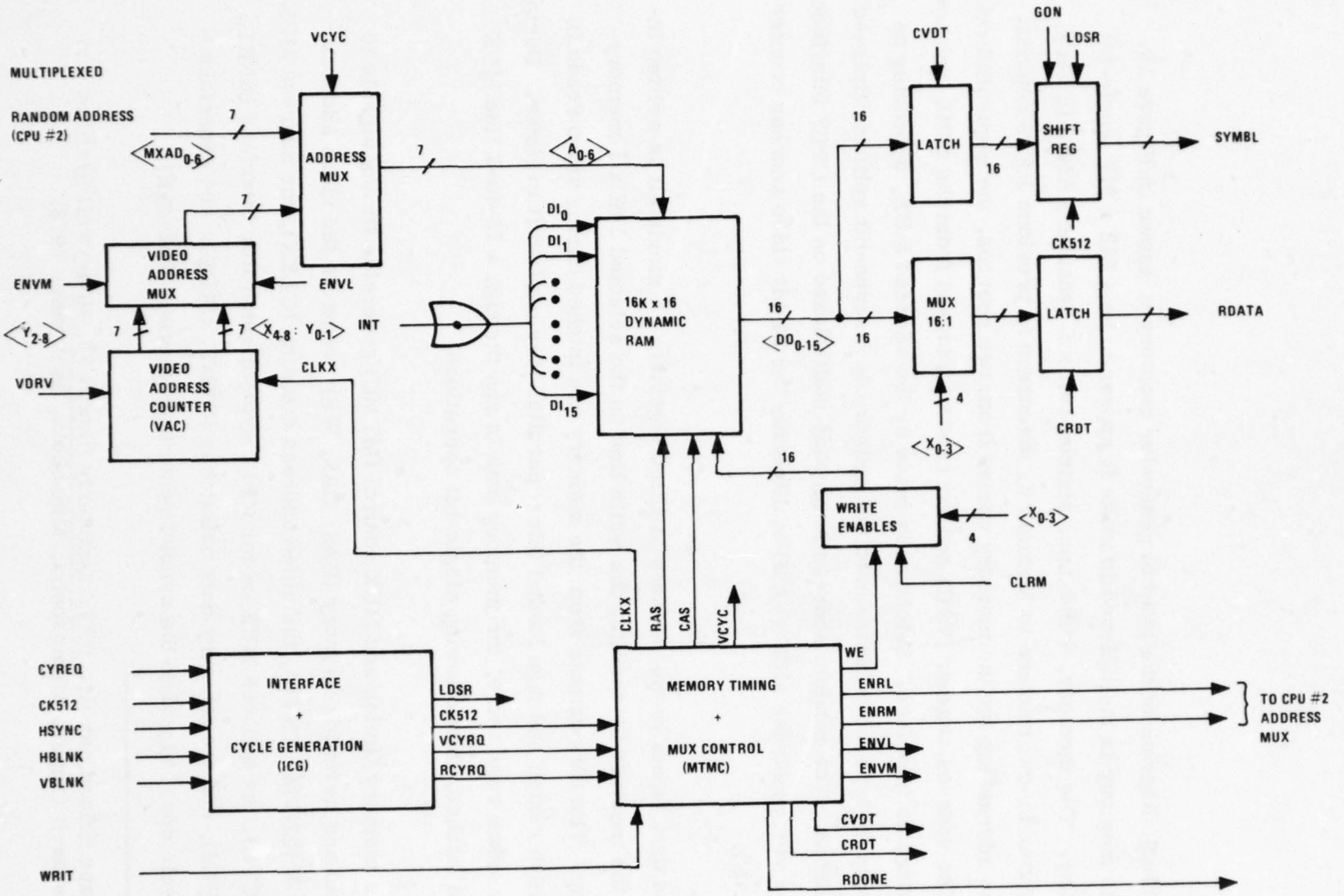


Figure 18. Symbol Generator Memory

The MTMC is clocked by the interface and cycle generation (ICG) section, which determines when a sequential cycle occurs and when a random cycle occurs. The ICG processes requests to read or write to memory from the CPU2 symbol generator interface (CYREQ) and synchronizes the requests to the clocking signals from the sync and timing board (CK512, HSYNC, HBLNK, VBLNK). The symbol output shift register is also controlled by the ICG (LDSR, CK512).

During vertical blanking (VBLNK), the VAC is cleared by the vertical drive signal (VDRV) and video reading is disabled. During this period, random cycles can occur every eight periods of the 512-clock from sync and timing (CK512), which is 500 ns for 875-line video and 800 ns for 525-line video. When vertical blanking is done, the video and random cycles alternate every eight periods of CK512 during active video. During horizontal blanking (HBLNK), random cycles can occur as during vertical blanking, except at the end of HBLNK when the alternating cycles begin again.

Timing diagrams for ICG signals at various points in the horizontal scan are shown in Figures 19, 20, and 21. In Figure 19, the HSYNC signal from the sync and timing board goes high sixteen CK512 periods before HBLNK ends. This prevents any random cycles (RCYRQ) from beginning. The HSYNC signal lasts for eight CK512 periods. At the end of HSYNC, a video read cycle (VCYRQ) is initiated for eight clock periods. The shift register load signal (LDSR) goes high at the end of the video cycle. A random cycle can occur following the video cycle (if CYREQ was sent from the symbol generator interface) as is shown in Figure 20, which is a continuation of Figure 19. The numbers in the CK512 clock pulses correspond to the memory

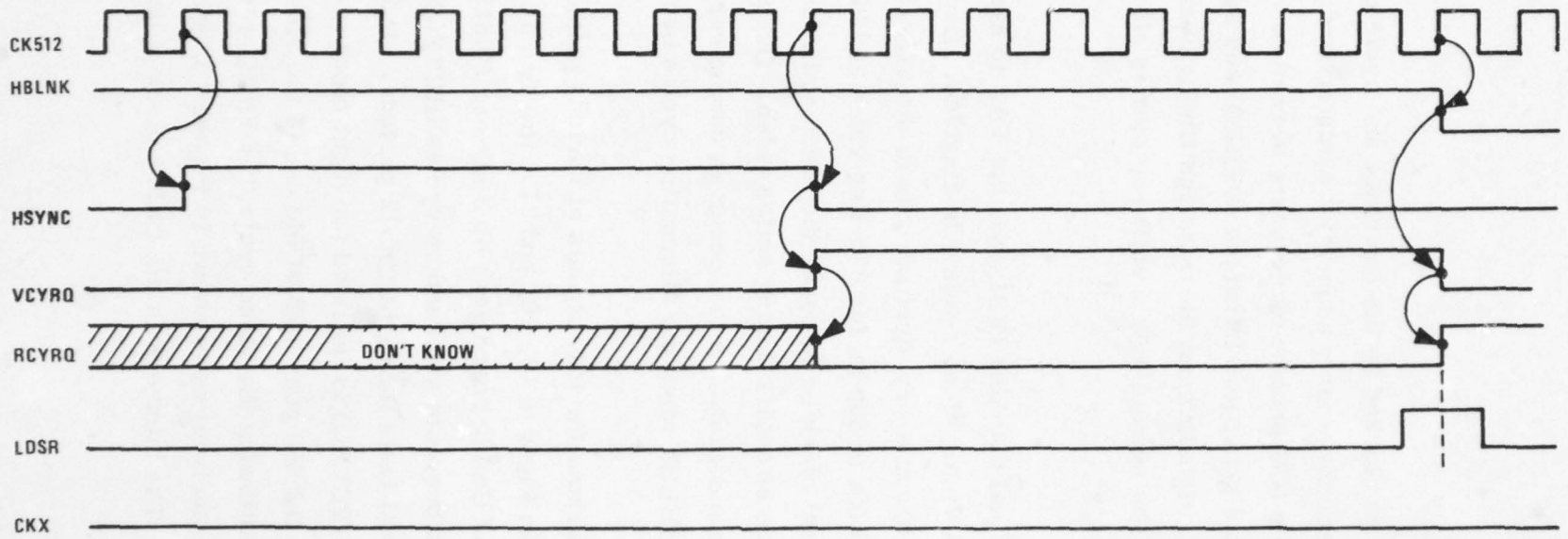


Figure 19. ICG Timing at the End of Horizontal Blanking

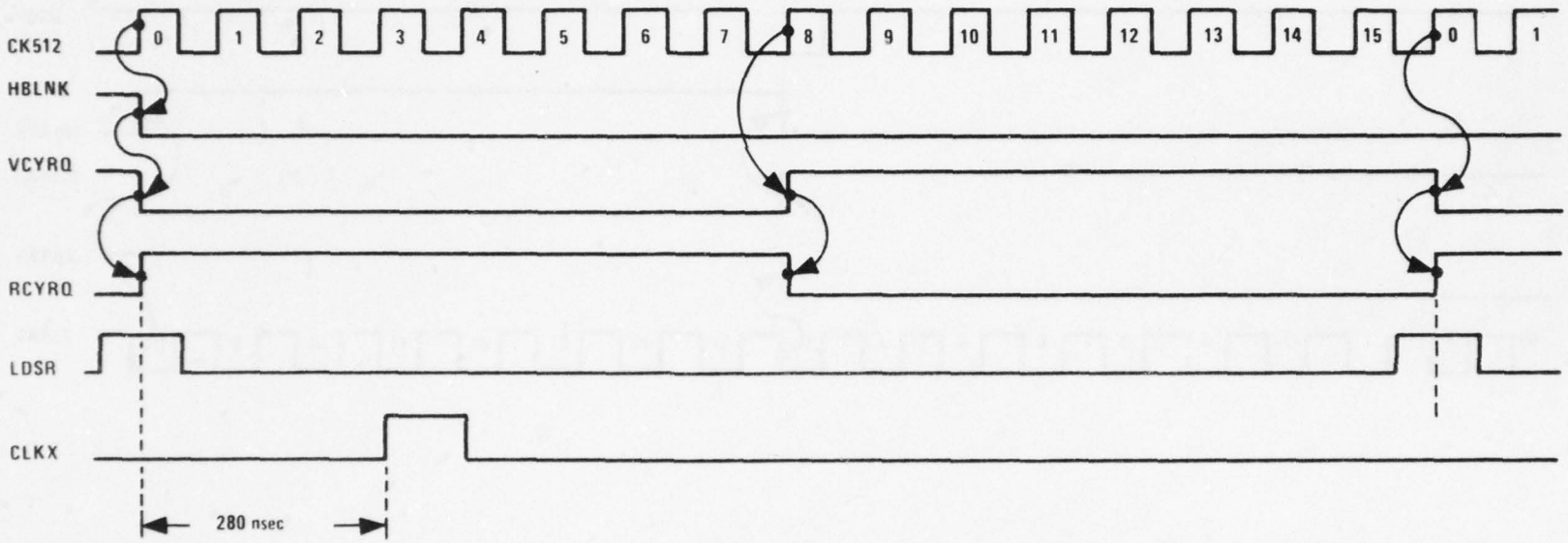


Figure 20. ICG Timing at the Beginning and During Video Scan Line

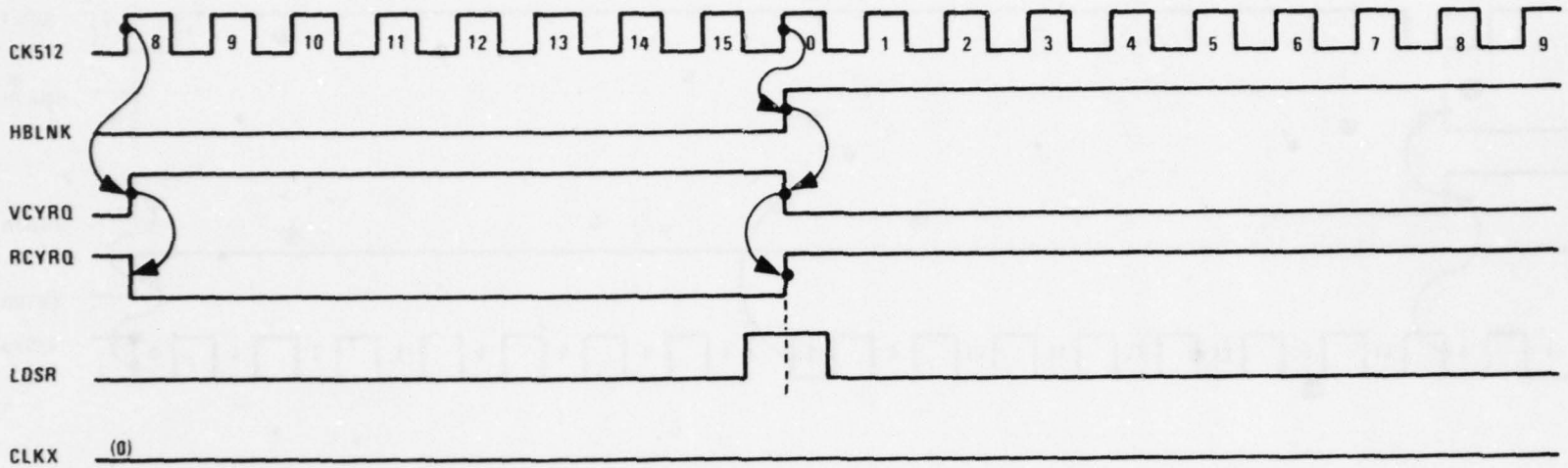


Figure 21. ICG Timing at the End of a Video Scan Line

data bit being shifted out of the shift register. During the random cycle, CLKX goes high, incrementing the VAC. This CLKX signal is gated by HBLNK to insure only 32 increments per line. A video cycle follows RCYRQ and alternates to the end of the video line. Figure 21 shows the timing at the end of a video line. After HBLNK goes high, all cycles until HYSNC goes high can be random (note that CLKX is low throughout blanking). This allows faster random access during blanking.

The timing diagrams for the signals to/from the MTMC are shown in Figure 22. The cycle request signals VCYRQ and RCYRQ dictate the actions of the MTMC. If RCYRQ, VCYRQ and HBLNK are low, the CLKX signal is only signal pulsing.

The cycle request (CYREQ), graphics on/off (GON), memory write (WRIT), clear memory (CLRM), and bit address (X_{0-3}) signals come from the CPU2 symbol generator interface. Signals to the interface are the address MUX controls (ENRM, ENRL) and cycle done (RDONE).

Vector Generator Implementation

An implementation block diagram of the vector generator is shown in Figure 23. Data to and from the LSI 11/2 comes over a 16-bit bi-directional data bus BD_{0-15}. The X and Y random address values are stored in nine-bit loadable up/down counters. Data from BD_{0-8} are loaded into the X counter when SELO and WCLKO are true. The counter is incremented or decremented by the CLKX signal, and XUD determines up or down count. The Y address counter behaves similarly. The concatenated results of X_{4-8} and Y_{0-1} from the counters form the least significant seven-bit multiplexed random address

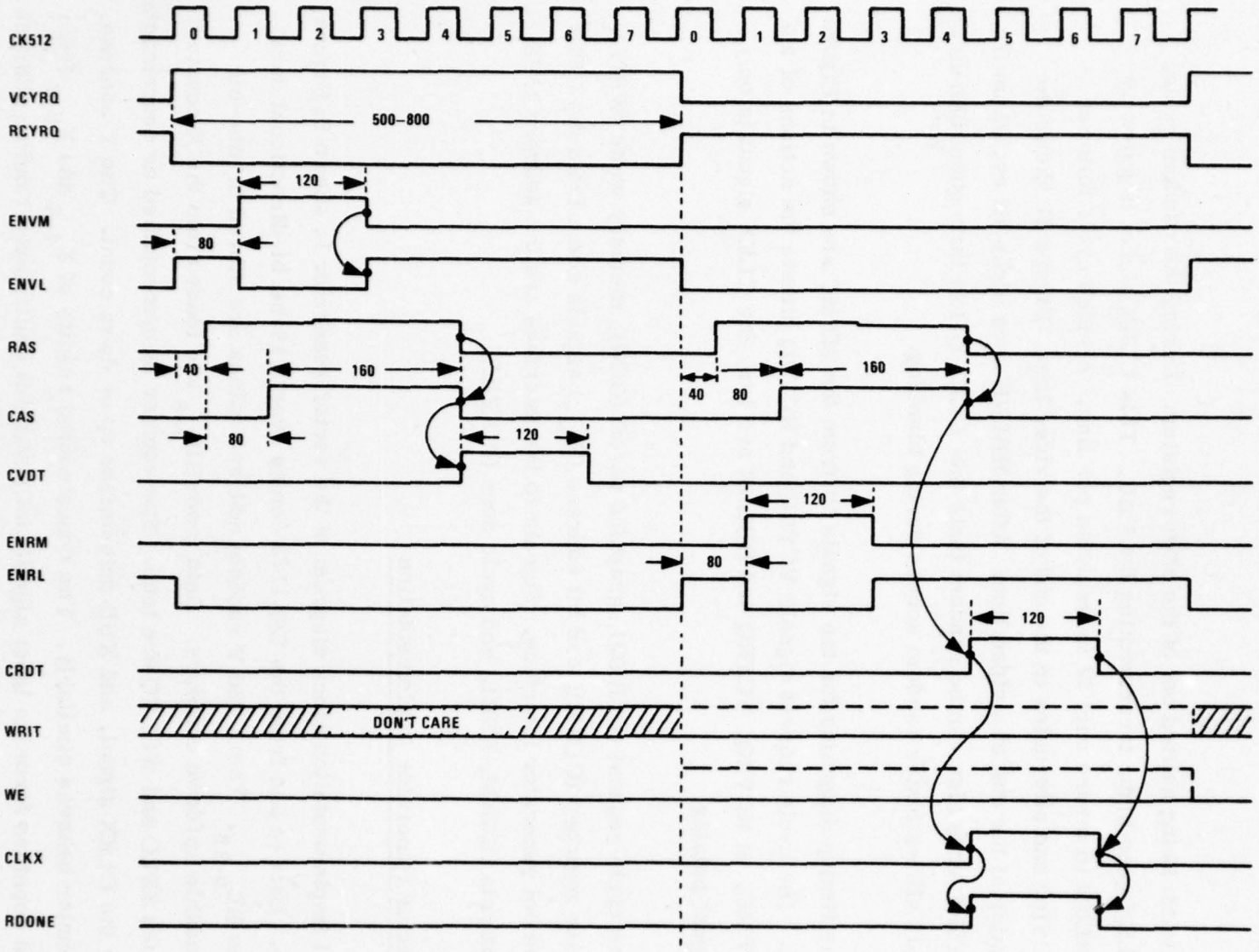


Figure 22. MTMC Timing Diagram

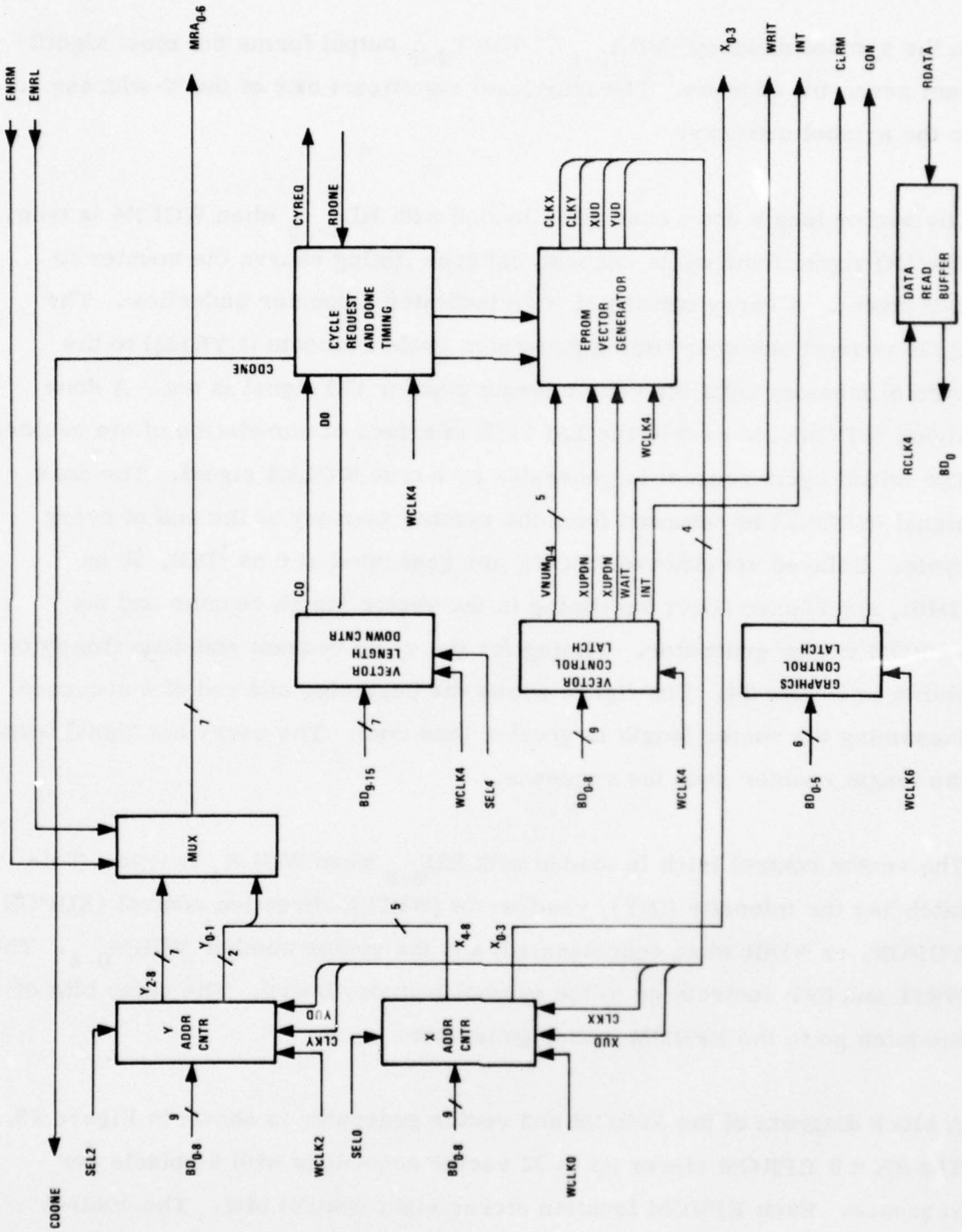


Figure 23. Vector Generator

to the symbol memory (MRA_{0-6}). The Y_{2-8} output forms the most significant seven-bit address. The four least significant bits of the X-address go to the symbol memory.

The vector length down counter is loaded with BD_{9-15} when WCLK4 is true. The DO signal from cycle request and done timing causes the counter to decrement. A carry out signal (CO) indicates a counter underflow. The cycle request and done timing generates cycle requests (CYREQ) to the symbol memory until the vector length counter CO signal is set. A done signal (CDONE) is sent to the LSI 11/2 interface at completion of the vector. The initial cycle request is generated by a true WCLK4 signal. The done signal (RDONE) is returned from the symbol memory at the end of every cycle. Delayed versions of RDONE are generated at 0 ns (DO), 40 ns (D40), and 80nsec (D80) for timing in the vector length counter and the EPROM vector generator. Timing for the cycle request and done timing is shown in Figure 24. The figure shows the beginning and end of a sequence (assuming the vector length is greater than one). The carry out signal from the length counter ends the sequence.

The vector control latch is loaded with BD_{0-8} when $WCLK_4$ is true. This latch has the intensity (INT), read/write (WRIT), direction control (XUPON, YUPON, or VDIR when concatenated) and the vector number $VNUM_{0-4}$. The WRIT and INT controls go to the symbol memory board. The other bits of the latch go to the EPROM vector generator.

A block diagram of the SPROM and vector generator is shown in Figure 25. The 2K x 8 EPROM stores up to 32 vector sequences with 64 pixels per sequence. Each EPROM location stores eight control bits. The control

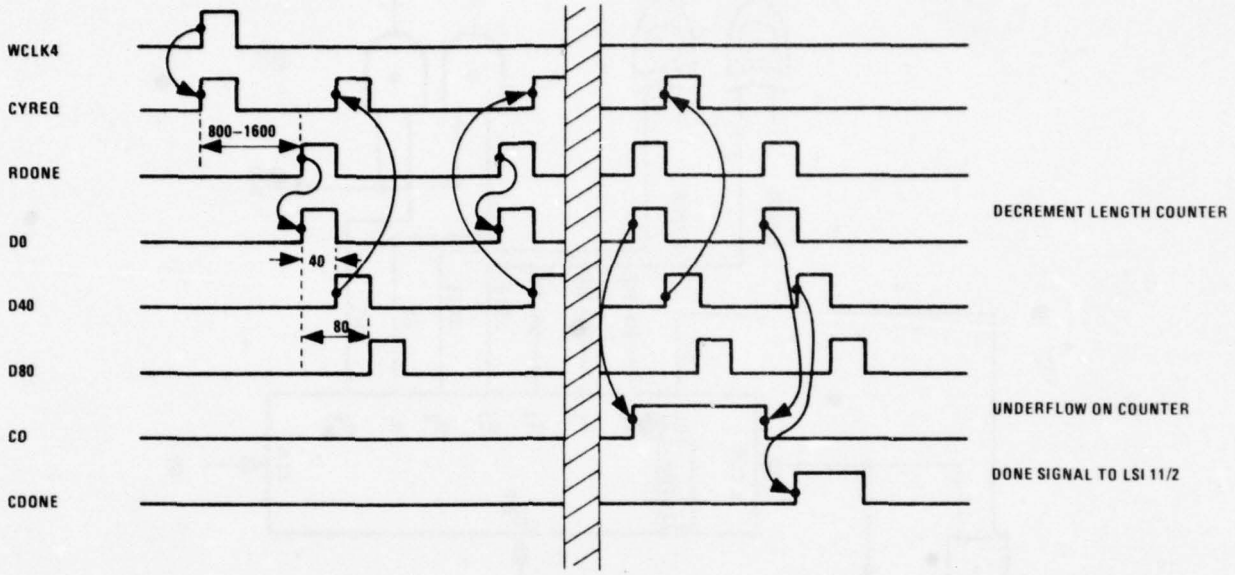


Figure 24. Vector Generator Cycle Request and Timing

bits determine the direction of the X and Y address counters and whether these counters are clocked during each symbol memory read or write cycle. The definitions of the control bits are shown in Table 3.

When WCLK4 is set, vector generation begins. As mentioned earlier, SCLK4 being set loads the vector control latch with XUPDN, YUPDN, and $VNUM_{0-4}$. The SCLK4 signal also clears the five-bit counter and the output latch on the EPROM. Thus, the least significant bits of the EPROM address (A_{0-5}) are cleared and the most significant bits of address (A_{6-10}) are $VNUM_{0-4}$. When RDONE returns from the symbol generator, the DO signal latches the EPROM contents. This sets the address counter directions XUD and YUD to their current or reverse direction with the YDIR and XDIR

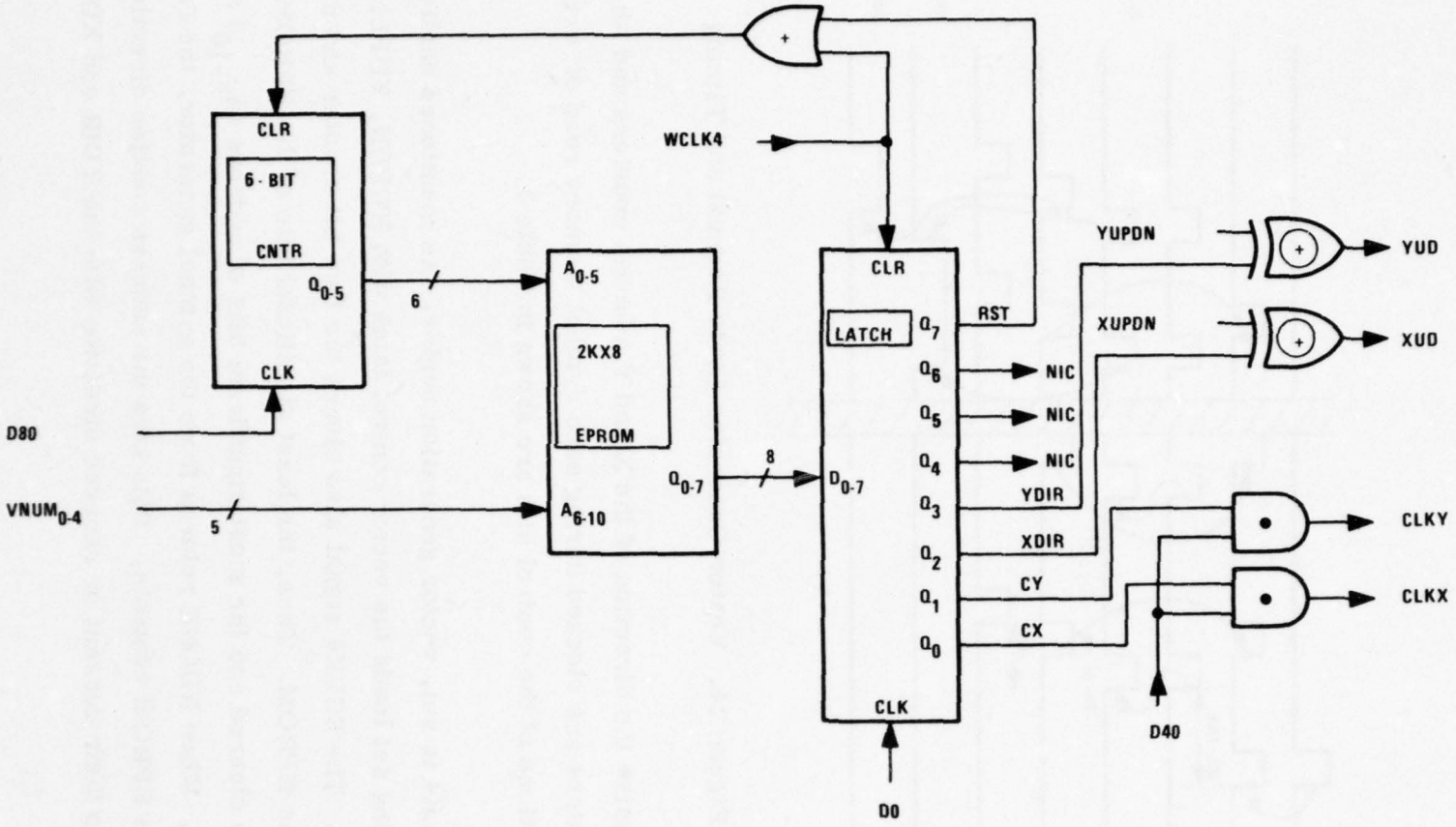


Figure 25. EPROM Vector Generator

TABLE 3. EPROM CONTROL BIT DEFINITION

EPROM Bit #	Symbol	Description
0	CX	CX = 1, clock X-address counter = 0, don't clocking X-address
1	CY	CY = 1, clock Y-address counter = 0, don't clock Y-address
2	XDIR	XDIR = 1, reverse counter direction specified by EUPDN = 0, counter direction set by XUPDN
3	YDIR	YDIR = 1, reverse counter direction specified by YUPDN = 0, counter direction set by YUPDN
4-6	---	NOT USED
7	RST	RST = 1, begin sequence over (reset 6 LSBS) = 0, continue to next address

signals from the latches using the exclusive OR gates. When D40 goes high, CLKX is set if CX is high and CLKY is set for CY high. If the RST signal is high, the six-bit address counter is cleared and the sequence is repeated. If RST is low, the counter is incremented by D80 to the next address in the EPROM.

Since the counter length is six bits, the sequence can be up to 64 bits long. The vector length counter underflow stops the EPROM vector generator. A

sequence in the PROM for $VNUM_{0-4} = 00100_2$ and a slope of 2/3 (See Figure 27) is shown in Table 4.

Programming Considerations for the Symbol Generator

Any bit within the symbol memory can be read. The X-address points to pixels along the video scan line and the Y-address to the scan lines. There are 480 active scan lines in the 525-line format. There are two lines displayed for every line in the graphics memory, so that only the first 240 locations of the graphics memory are displayed. For the 875-line rate, 400 locations in the memory are used. Any bit within the memory can be read or written. This can be done at all times, but occurs faster during the video blanking periods. Fastest access to the memory occurs for 1.3 msec after the video vertical retrace period begins. A signal is returned to the LSI 11/2 to indicate that this period has started (VDRV interrupt). The maximum rate

TABLE 4. EPROM SEQUENCE FOR SLOPE 2/3

EPROM ADDR											EPROM CONTENTS							
10	9	8	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	0	1	0	0	0	0	0	0	0	0	0	-	-	-	0	0	1	1
0	0	1	0	0	0	0	0	0	0	1	0	-	-	-	0	0	1	0
0	0	1	0	0	0	0	0	0	1	0	1	-	-	-	0	0	1	1
0	0	1	0	0	ALL OTHERS						-	-	-	-	-	-	-	-

at which the memory can be accessed continuously is 2 MHz/bit during vertical retrace and at least 1 MHz/bit during the remainder of the video field.

The memory is organized as a 512 x 512 array of single bit words. Each word in the memory is addressable by a 9-bit X-address and 9-bit Y-address. The addresses correspond to pixels on the screen of a faster scan video monitor as shown in Figure 26.

The vector generation feature of the interface allows the user to specify a starting X, Y position and draw fixed slope vectors of variable length. The parameters necessary to generate the vector, in addition to the starting address, are: (1) the vector type (in other words, usually slope), (2) the quadrant that the vector is drawn in using the origin as the starting X, Y

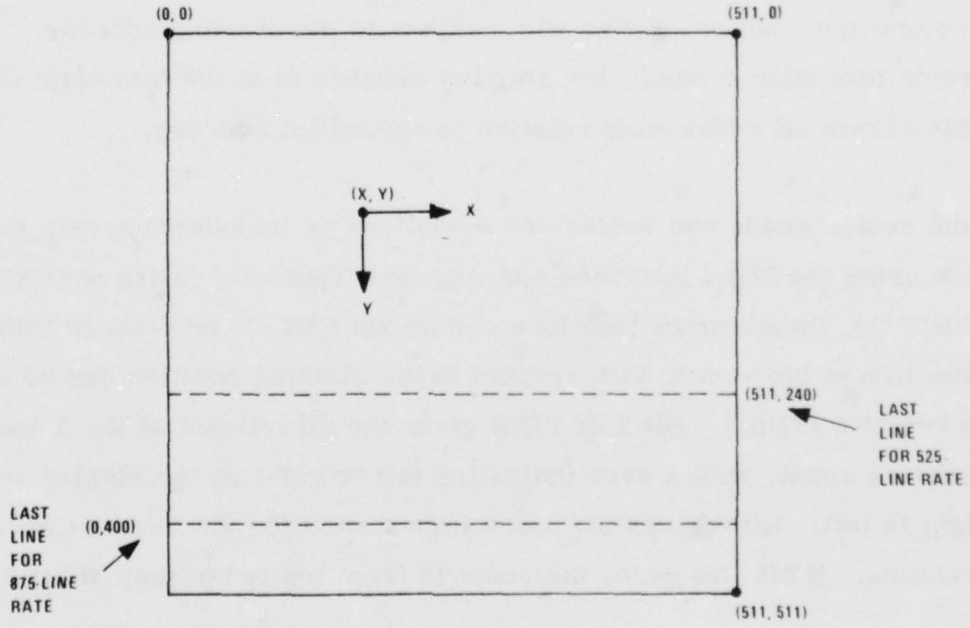


Figure 26. Graphics Display Formats

address, (3) the length of the vector in pixels, and (4) whether the vector is erased or intensified. Once these parameters are specified, a vector is automatically generated allowing the computer to do other functions. Vector slopes have not yet been selected, but typical values for Y/X might be 1, 1/2, 1/3, 2, 3, etc. This method of vector generation does not allow general vector generation as a $\Delta Y/\Delta X$ approach would, but should be adequate for most applications. Slight modification of the approach allows curve generation in addition to vectors. Single points are written into the memory by specifying a unit length vector.

The format of the symbol generator commands is as follows:

Starting addresses in the graphics memory are set up by issuing X- and Y-address values to the interface addresses SELØ (X-address) and SEL2 (Y-address). The format of the data to the interface is shown in Table 5. Vectors are read or written with respect to the starting address. After a vector operation is done, the starting address is at the last point of the vector. This allows all addressing relative to any initial address.

Both vector reads and writes are initialized by transferring data to the interface using the SEL4 interface address (see Table 6). When writing a vector (WRIT=1), the memory locations can be set (INT=1) or cleared (INT=0). The direction of the vector with respect to the starting position can be selected by two bits (VDIR). Bit 2 of VDIR gives the direction that the X location counters count, with a zero indicating left to right on the display and a one right to left. Bit 3 gives the same information for the Y-address location counters. If bit 3 is zero, the count is from top to bottom; if bit 3 is one,

TABLE 5. X AND Y ADDRESS COMMAND FORMATS (SELØ, SEL2)

SET X-ADDRESS

SELØ Output Data To Interface

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								X-ADDRESS							

X-ADDR — Nine bit starting X-address. Values from 0 (left) to 511 (right).

SET Y-ADDRESS

SEL2 Output Data To Interface

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED								Y-ADDRESS							

Y-ADDR — Nine bit starting Y-address. Values from 0 (top) to 511 (bottom).

TABLE 6. VECTOR READ/WRITE COMMAND FORMAT (SEL4)

SEL4 Output

Data To Interface

15 14 13 12 11 10 9	8 7 6 5 4	3 2	1	0
VLEN	VNUM	VDIR	WRIT	INT

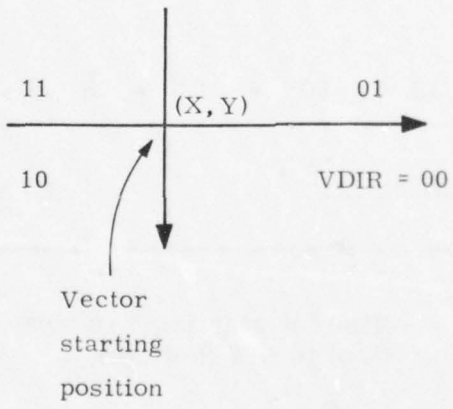
INT = 1 Intensify vector if WRIT = 1

= 0 Erase vector if WRIT = 1
If WRIT = 0, then INT causes no-operation

WRIT = 1 Write vector

= 0 Read vector

VDIR = Vector direction from starting point.



VNUM - Vector number from 0-31 which selects one of the preprogrammed vectors.

VLEN - The length of the vector in pixels, up to 128 elements.

the count is from bottom to top. The concatenated value of the 2 bits gives a vector quadrant as shown in Table 6 under the VDIR description. The vector type (or slope), specified by VNUM, allows for 32 different angles and, combined with the quadrant specification, gives 128 possible angles. A discussion of the angles will be provided later. The vector can be up to 128 pixels in length (VLEN). If a single location is to be read or written, a length of one is specified.

Reading a vector does not modify the contents of the graphics memory, so that reading simply moves the current position address to another location. The value of the graphics memory location at the end of the vector that was read can be loaded into the computer by moving data from the interface using the SEL4 address. The data comes back in bit Ø as shown in Table 7.

TABLE 7. READ VECTOR DATA COMMAND FORMAT (SEL4)

SEL4 Input	Read Vector Data															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data To Interface	NOT USED															DATI

DATI - Value of graphics memory at the last point in a vector that was read.

The graphics mode can be controlled by moving data to the SEL6 address of the interface. The format of the data is shown in Table 8. Graphics can be displayed by setting bit Ø (GON=1). The entire graphics memory can be cleared by setting bit 1 (CLRM=1). It takes video field time 1/60 second to clear the memory. When bit 2 is clear (ALT=0), the graphics display produces an intensified pixel (white) when the graphics memory bit is a one. When the memory bit is a zero, only the video appears on the display. When bit 2 is set (ALT=1), the pixel will be half-white/half-black when the graphics memory bit is set.

Vectors in raster scan graphics are generated by incrementing or decrementing X- and Y-address counters. Slopes of vectors are rational fractions. An example of a vector generated with a slope of X/Y = 2/3 is shown in Figure 27.

TABLE 8. GRAPHICS CONTROL COMMAND FORMAT (SEL6)

SEL6 Output — Data To Interface

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED										RESERVED			ALT	CLRM	GON

GON = 1 Display graphics memory
= 0 Disable display of graphics memory

CLRM = 0 No operation
= 1 Clear entire graphics memory

ALT = 0 Display graphics bit set as white on the display(0)
= 1 Alternate white and black during every pixel that a graphics bit is set(1)

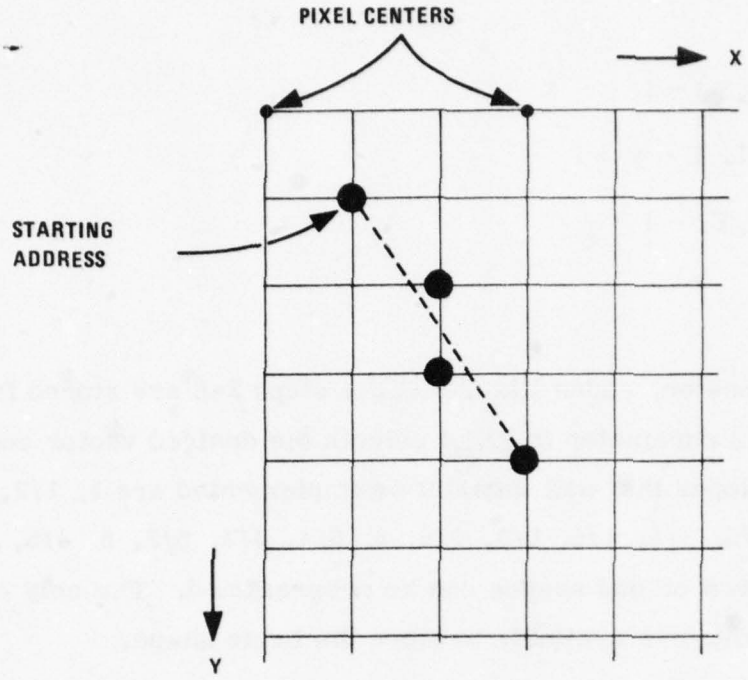


Figure 27. A 2/3 Slope Vector of Four Points

The dashed line shows an ideal vector, but since a raster scan device has discrete points, the vectors are seldom straight. Usually, a least-squares approximation to the vector is used. This is shown by the dots in Figure 27.

To generate the vector, the following sequence of operations must occur.

1. $X \leftarrow X\emptyset$, $Y \leftarrow Y\emptyset$ initialize X- and Y-address
2. MEM [X,Y] $\leftarrow$ 1
3. $X \leftarrow X + 1$, $Y \leftarrow Y + 1$
4. MEM [X,Y] $\leftarrow$ 1
5. $Y \leftarrow Y + 1$

6. MEM [X,Y] ← 1

7. X ← X + 1, Y ← Y + 1

8. MEM [X,Y] ← 1

9. END

In the vector generator, codes like the above steps 2-8 are stored for each slope. The VNUM parameter in SEL4 selects the desired vector code sequence. The slopes that will initially be implemented are 1, 1/2, 1/3, 1/4, 3/4, 1/5, 2/5, 3/5, 4/5, 1/6, 5/6, 6, 5/4, 5/3, 5/2, 5, 4/3, 4, 3/2, 2. In addition, arcs or odd shapes can be programmed. The only constraint is that only 64 points are available to store the basic shape.

The basic slopes are shown in Figure 28. Note that some slopes have very close angles (4/5 and 5/6). It may be desirable to eliminate one of these. An example of an arc is shown in Figure 29. Odd shapes can also be generated as shown in Figure 30.

DMA BOARD

A DMA channel is required that allows for DMA access to the 11/2 for 120 (variable) words with programmed I/O occuring on CPU1. The DMA L-11 board meets most of the requirements for the LSI 11/2 DMA interface. A block diagram of the DMA-L11 is shown in Figure 31. Data transfer will be by word mode only. The DMA-L11 may operate as programmed I/O or direct memory access I/O.

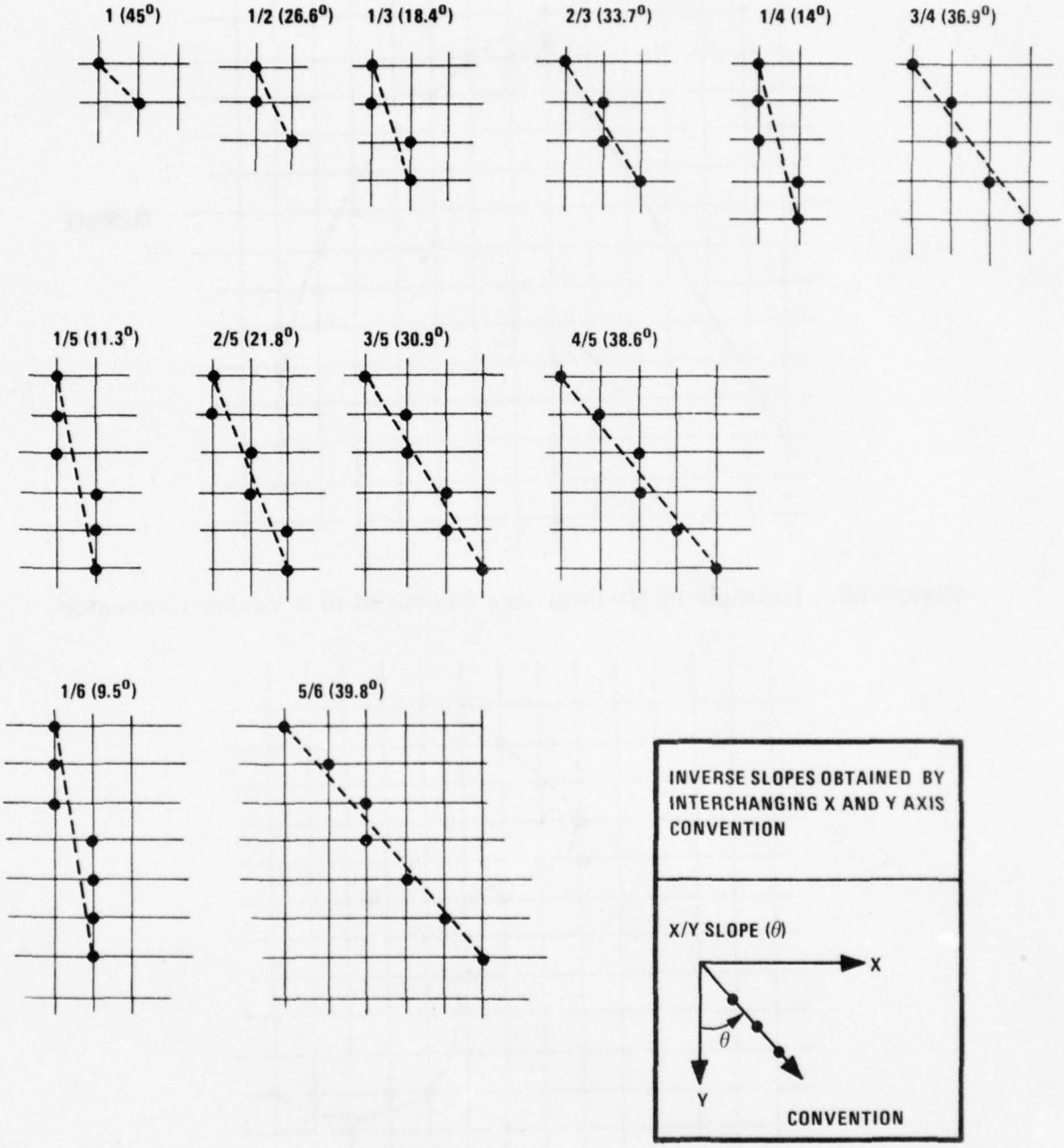


Figure 28. Sample Vector Slope Plots

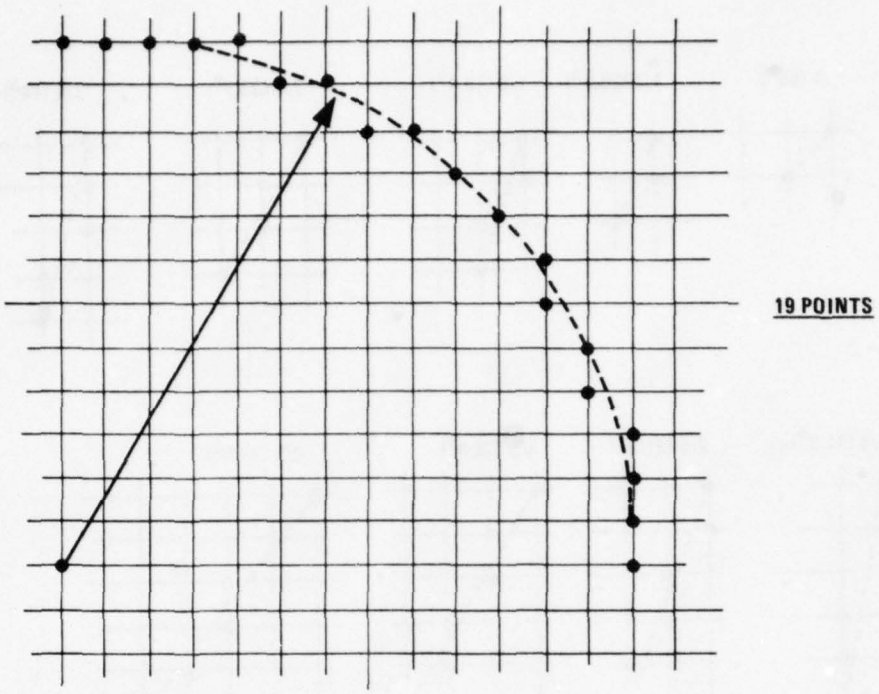


Figure 29. Example of 90-Deg. Arc Produced in a Vector Generator

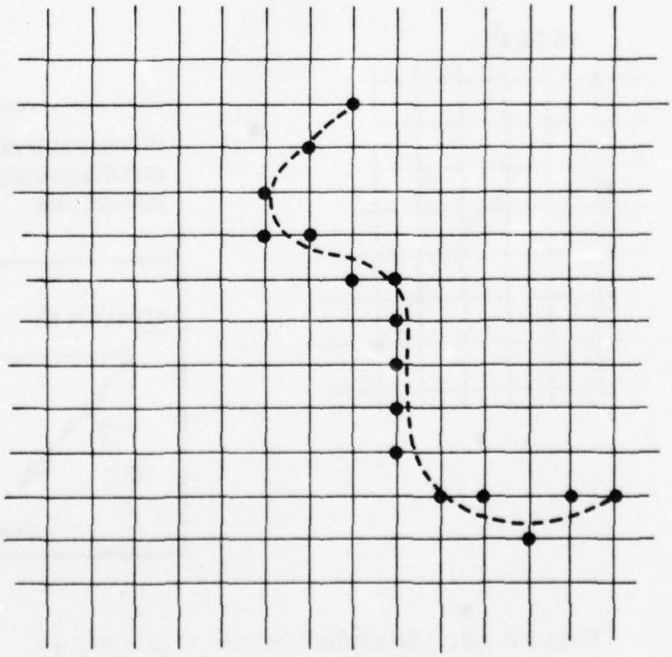


Figure 30. Example of an Odd Shape Generated by the Vector Generator

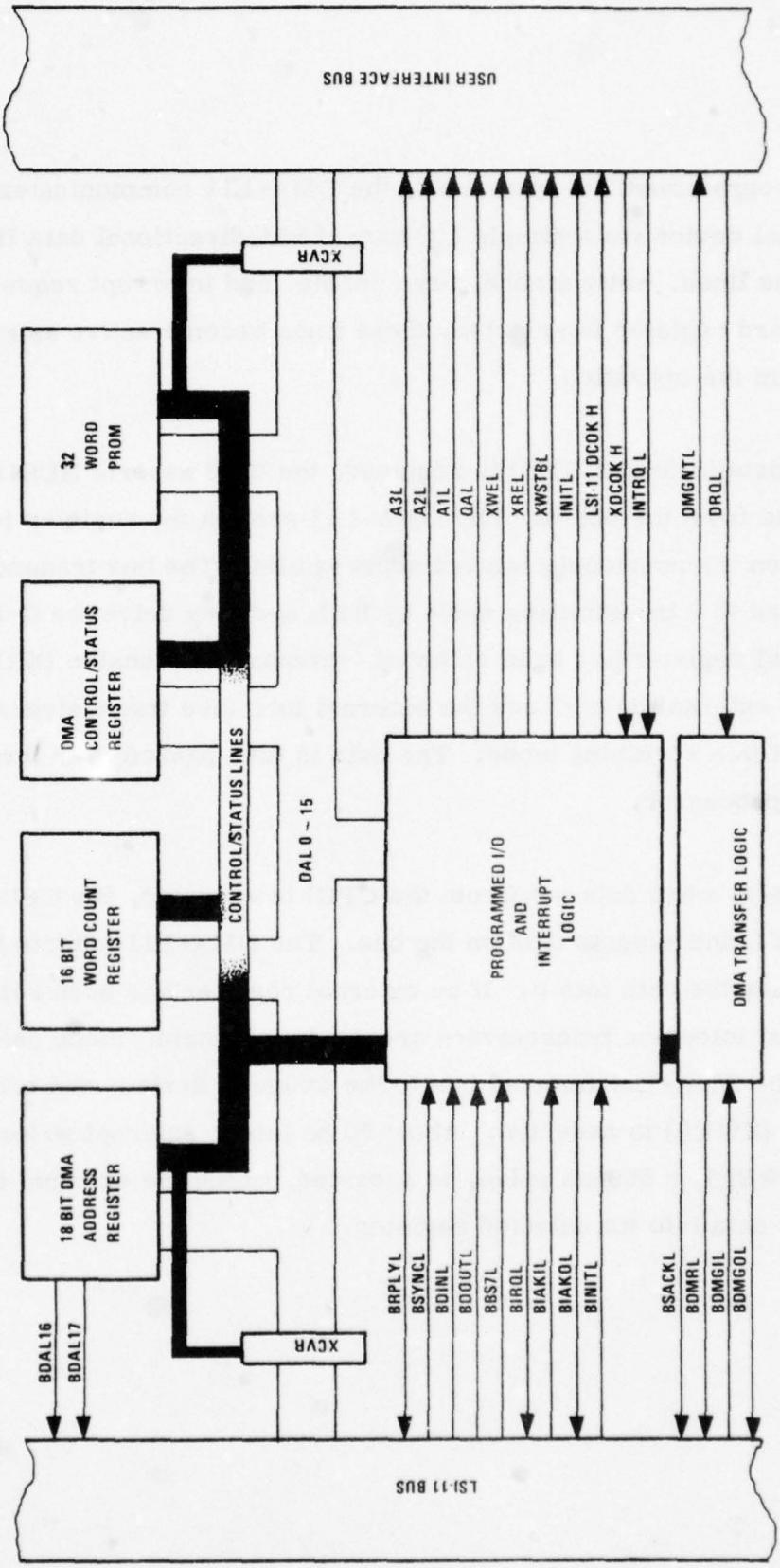


Figure 31. DMA-L11 Interface

For programmed I/O operations, the DMA-L11 communicates with the external device via a simple I/O bus; 16 bi-directional data lines, three address lines, write strobe, read enable, and interrupt request. When an off-board register is selected, these lines become active as required to perform the operation.

When data in (to the CPU) is required, the CPU asserts BDINL and waits for data from the board. The DMA-L11 selects the register (or PROM) based on the previously latched address bits. The bus transceivers are switched to a transmitting mode by REL and they drive the Q-bus. If an external register has been selected, external read enable (XREL) is asserted (to the external device) and the external interface transceivers (normally off) are put in a receiving mode. The data is then passed over the internal bus to the processor.

Similarly, when data out (from the CPU) is executed, the CPU asserts BDOUTL and asserts data on the bus. The DMA-L11 selects the register and loads the data into it. If an external register has been selected, the external interface transceivers are put in a transmit mode and pass data from the DMA-L11 internal bus to the external device, and external write enable (XWEL) is asserted. About 50 ns later, external write strobe (XWSTRBL), a 350-ns pulse, is asserted, which the external device can use to load data into its selected register.

In the case of data either in or out, the DMA-L11 terminates the cycle by asserting BRPLYL. The processor examines the state of BRPLYL every 350 ns and, when it finds it asserted, releases BDINL or BDOTL, completing the bus operation. The assertion of BRPLYL is delayed to permit external devices with an access time of up to 450 ns to be operated reliably. This time may be changed such that the length of BDOUTL or BDINL can be set to be between 350 ns and 10 μs after the I/O cycle has started.

Interrupt

The DMA-L11 provides the system with a flexible interrupt facility. There is only one interrupt vector address, but the source of the interrupt may be from either of two classes: internal or external, with a separate enable for each class. An external interrupt comes from the external interface interrupt request input line, INTROL. An internal interrupt can come from byte count overflow, external DC power failure, or (as a jumper option) bus timeout. It should be noted that if any of the interrupt source lines is active when an interrupt enable for that class is set, an interrupt request will be generated; also, if an interrupt has been asserted and cleared, with processor interrupts disabled and DMA-L11 interrupt enabled, an interrupt will occur when processor interrupts are enabled. These conditions may best be avoided by making sure an interrupt is not pending when enabling interrupts or, alternately, one may write into the DMA memory address register, which will also clear the interrupt flipflop. One may ascertain the status of the interrupt lines by examining the DMA control/status register.

When interrupts are enabled and interrupt has been requested by one of the sources heretofore mentioned, the DMA-L11 asserts BIROL; thereupon, any CPU DIN cycle will latch the interrupt into the interrupt acknowledge flipflop. When the CPU services the interrupt, it asserts BDINL and BIAKL (which the DMA-L11 would pass to the next device on the bus if it had not requested an interrupt). The DMA-L11 drives the BDAL lines with the interrupt vector address, BRPLYL is asserted, and the interrupt request flipflop is cleared. When the CPU receives the interrupt vector address and BRPLYL it discontinues BIAKIL and BDINL. It then executes a DIN cycle, when reading the contents of memory at the vector address location, which clears the interrupt acknowledge flipflop.

When the external device requests DMA transfer, the rising edge of DRQ sets the DMA request flipflop (if the byte count is not at 177777), increments the address and byte counters, and requests a DMA cycle from the processor. If the CPU is busy, the grant will be delayed until the instruction has been completed; when it does issue BDMGIL, the negation of BSYNCL and BRPLYL by the CPU clocks the DMAGNT flipflop. The DMAGNT flipflop than asserts BSACKL, which tells the processor that the DMA-L11 has assumed bus mastership. DMAGNT is then latched into the DMASYNC flipflop, which synchronizes the DMAGNT to the DMA-L11 clock. The output of DMASYNC causes the DMA address counter contents to be asserted on the Q-bus by enabling the register outputs and timing on the bus transceivers with DMADENL, clears the DMAREQ flipflop, and notifies the external device that DMA requests has been granted. BWTBTL is asserted at this time if a DOUT cycle is to be performed. The signal will be negated at the onset of the I/O phase if word transfers are to take place; otherwise, it remains

asserted until the cycle is complete. 279 ns later, BSYNCL is asserted, which loads all devices on the Q-bus with the contents of the DMA address register. BSYNCL is maintained for the duration of the DMA transfer cycle. 279 ns after BSYNCL has been asserted DMADENL is negated and disables the address outputs. The timing now enters the I/O phase of the transfer.

When a bit 5 of the DMA control/status register is set, it indicates that a write to the external device is desired. This means that the DMA-L11, as bus master, will perform a DIN cycle. Three clock periods are required to complete the transfer. In the first cycle, BDINL is asserted, whereupon the selected memory should respond with data and BRPYLY. The second clock period checks for reception of BRPLYL. If it has not been received by the end of the period, then the DMA-L11 hangs in this wait state until it does come. If BRPLYL does not come within 10 μs, the transfer is aborted, the BUSERR flipflop is set and further data transfers are inhibited. The BUSERR flipflop is cleared by writing into the DMA address register. During the third period XWEL is asserted, which transmits the data from memory to the user device by enabling the external interface bus transceivers, and XWSTBL, which can be used to strobe the transmitted data into an external register. XWSTBL should be used for this function, since it ensures reasonable data setup and hold times for the data. With the rising edge of the fourth period, the transfer is complete and BSYNCL, BSACKL, BDINL and XWEL are negated.

When bit 5 of the DMA control/status register is zero, it indicates that data is to be read from the external device into memory. The DMA-L11, as bus master, executes a DOUT cycle. Three clock periods are required to complete

this part of the transfer. In the first period, XREL is asserted to the external device and causes the external interface transceivers to receive data from the user device and the internal bus transceivers to drive the Q-bus with that data. The user device must supply valid data within 450 ns after XREL has been asserted or else data will not be set up long enough on the Q-bus to satisfy the Q-bus specification. In the second period BDOUTL is asserted, which informs the addressed device that valid data is on the bus. At the start of the third period BRPLYL is checked to see if it has been asserted; the timing will hang for no more than 10 μs in this state in expectation of BRPLYL; if this time limit is exceeded the transfer is aborted, the BUSERR flipflop is set, and further DMA transfers are inhibited until the BUSERR flipflop has been cleared. When BRPLYL is properly received, BDOUTL is negated about 200 ns after the rising edge of the clock. There is than a 350-ns data hold time for memories which load data on the trailing edge of BDOUTL. The DMA cycle is complete with the rising edge of the fourth clock period and BSYNCL, BSACKL, BWTBTL (if a byte transfer has been executed), and XREL are cleared.

There is no distinction made between burst or single transfer mode by the DMA-L11. It will request DMA transfers whenever requests are received from the external device. In fact, a DMA request may be pending while a DMA transfer is in progress. However, the external device must wait at least one master clock period (558 ns) after XDMAGNT has been secured before asserting DRQ again. This is because DRQ is prevented from setting the DMAREQ flipflop while the DMA address is being presented to the Q-bus (since setting the DMAREQ flipflop increments the DMA address and byte count registers).

Programming Considerations

For programming purposes, the following characteristics of the DMA-L11 apply:

17740	Base Address
200	Interrupt Vector Address
177500	External Device Register 0
177502	External Device Register 1
177504	External Device Register 2
177506	External Device Register 3
177510	External Device Register 4
177512	DMA Control/Status
177514	DMA Memory Address
177516	DMA Byte Count

The on-board registers must be set up properly before a DMA transfer operation can be started for the DMA control/status register. The bits have the following meaning:

Bit 15	R	External device interrupt request
	W	External device interrupt enabled
14	R	Internal interrupt request
	W	Internal interrupt enabled
12	R	Bus error (timeout)
7	R	XDCOK (high if external power low)
5	W	DMA W/R (high for DMA write to external device)
1	R/W	Extended address bit 17
0	R/W	Extended address bit 16
13	W	Wux H (signal line to external device) (opticnal)

All unused bits in the control/status register read as 1's.

Relative Address 114		DMA Address Counter
Bits 0-15	Read/Write	DMA address bits 0-15

A write operation into device address 114 will clear the bus timeout error flag and the interrupt request latch.

This counter will "wrap-around" to zero when it reaches maximum count and will increment the extended address bits at that time.

This counter is incremented by two 4-word mode transfers. The incrementing occurs before the counter contents are used as the address. Therefore, the value loaded into this counter must be less than the actual memory address where the first transfer is to occur. In word mode the memory address 2 should be loaded into this counter.

Relative Address 116		Word/Byte Counter
Bits 0-15	Read/Write	Word or Byte Count

This up-counter will inhibit further DMA transfers when it reaches the maximum count of 177777 and will generate an interrupt if internal interrupt is enabled.

This counter is incremented by one if byte mode transfer is selected and by two if word mode transfer is selected. Thus, it always counts bytes regardless of the transfer mode selected. This is an up-counter so it must be loaded with the negative byte count. Incrementing is done before each DMA transfer cycle so the actual value to be loaded into this counter is:

- (byte count)-1 for byte mode
- (byte count)-2 for word mode

Off-Board Registers

Five word locations are reserved for off-board registers. These registers may be up to 16 bits wide each, read and/or write. All data must be buffered by latches, as needed, because the DMA-L11 uses an external bi-directional data bus and does not latch data in or out of the board.

Device Address	100, 102, 104, 106, 110	Off-Board Registers
Bits 0-15	Read/Write	Registers assigned by user

Interfacing to CPU1

The following signal definitions apply to signals available on the DMA-L11 connector 40.

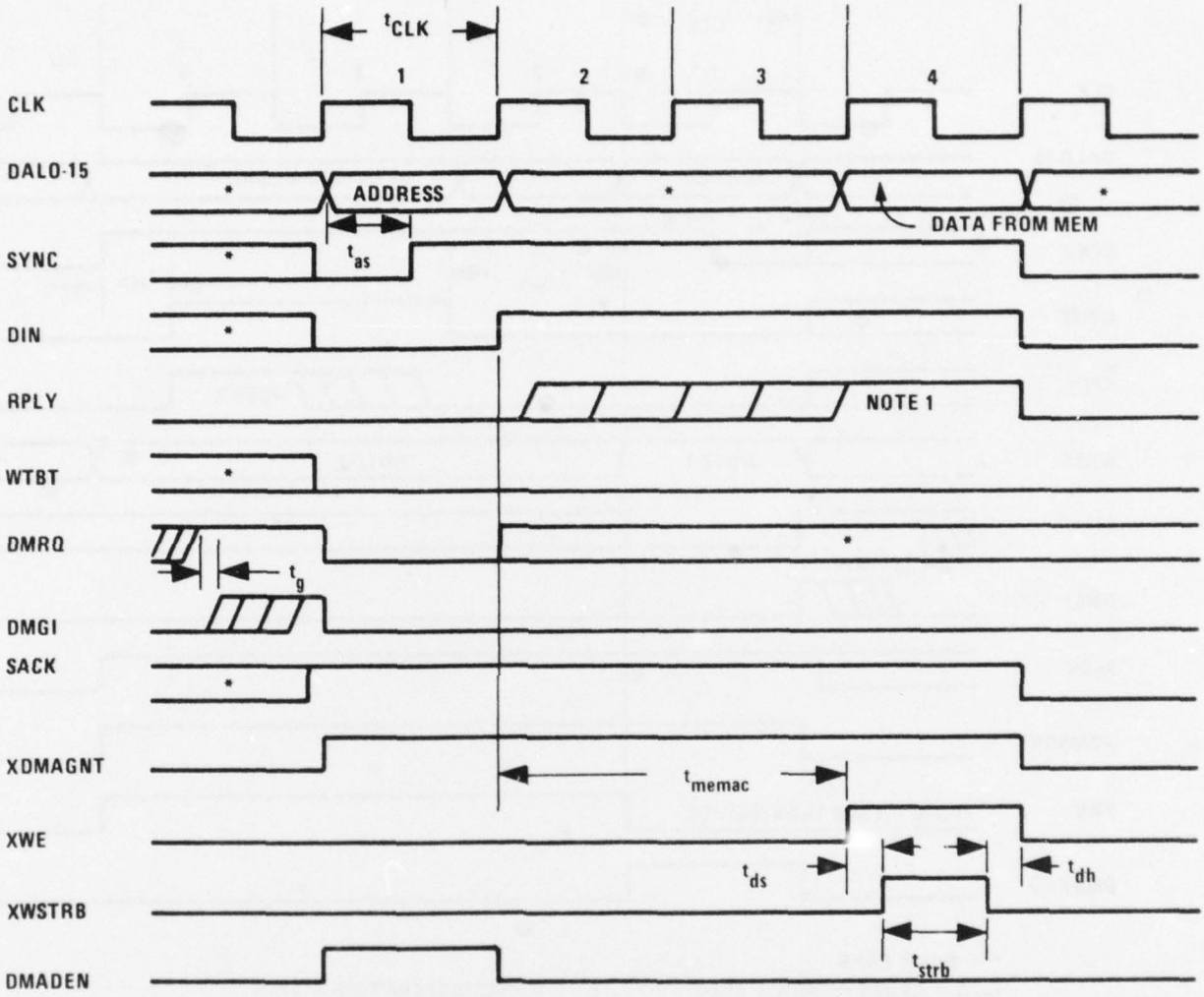
POW	SIGNAL	DESCRIPTION
VV	XDAL15L	Bi-directional tri-state data
TT	XDAL14L	bus normally in high
RR	XDAL13L	impedance state
NN	XDAL12L	
LL	XDAL11L	
JJ	XDAL1ØL	
FF	XDALØ9L	
DD	XDALØ8L	
CC	XDALØ7L	
EE	XDALØ6L	
HH	XDALØ5L	
KK	XDALØ4L	

MM	XDALØ3L	
PP	XDALØ2L	
SS	XDALØ1L	
UU	XDALØØL	

A	DCOKH	LSI 11/2 power OK
B	DRQL	DMA request asserted by CPU1
D	INTRQL	Interrupt request by CPU1
F	QAL	Low order DMA address bit (not used)
J	XDCOKH	CPU1 power OK
L	A3L	External register address lines (not used in DMA)
R	A2L	000=R0, 001=R1, 010=R2
V	A1L	011=R3, 1-0=R4 use with XREL or XWEL
N	DMAGNTL	DMA has been granted
T	TNITL	Initialize and clear
X	XWEL	Ext write enable, data to CPU1
Z	XREL	Ext read enable, data expected from CPU1
BB	XWSTBL	Ext write strobe
C,E,H,K,M,P,S,U, W,Y,AA		GND

All control lines should be terminated with 150 ohm pullup since they are driven with 740's. All inputs are terminated with 150 ohm pullups and 7414 Schmitt trigger inputs.

Figure 32 shows relative timing for DMA timing reading from memory, Figure 33 shows timing for DMA write into memory, and Figure 34 is the programmed I/O timing. Note that AXL signals are used only in programmed I/O.



* = DON'T CARE

t_{CLK} = CLOCK PERIOD = 558ns t_{as} = ADDRESS SETUP TIME = 279ns

t_g = DMA GRANT TIME = UNDEFINED – DEPENDENT ON CPU ACTIVITY

t_{memoc} = MAXIMUM MEMORY ACCESS TIME BEFORE TRIGGERING A WAIT IN T53 ≃1000ns

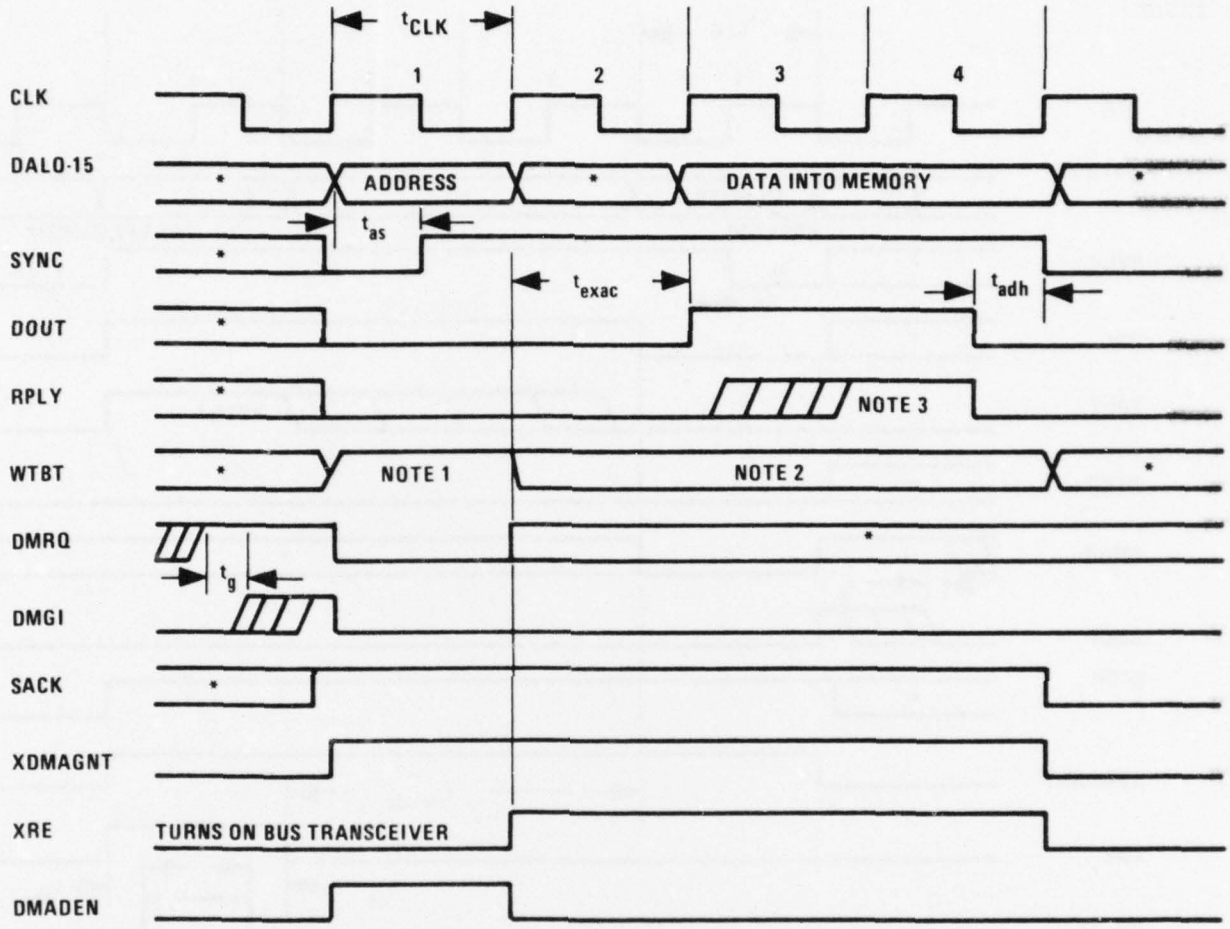
t_{as} = DATA SETUP TIME BEFORE STROBE ≃70ns

t_{strb} = WRITE STROBE PULSE WIDTH ≃375ns

t_{dh} = DATA HOLD TIME AFTER STROBE ≃110ns

NOTE 1: SEE NOTE 3 IN FIGURE 32

Figure 32. DMA-L11 DMA Timing: Write Into Memory



* = DON'T CARE

t_{CLK} = CLOCK PERIOD = 558ns t_{as} = ADDRESS SETUP TIME = 279ns

t_{exac} = EXTERNAL DEVICE ACCESS TIME $\leq$450ns

t_{odh} = OUTPUT DATA HOLD $\geq$ 250ns

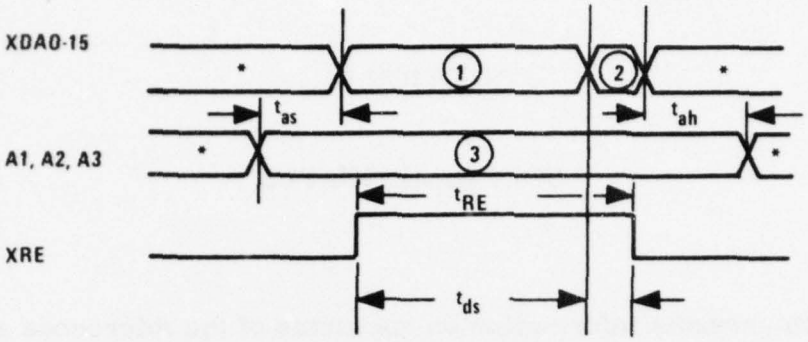
t_g = DMA GRANT TIME = UNDEFINED – DEPENDENT ON CPU ACTIVITY

NOTE 1: HIGH WHEN WRITE INTO MEMORY

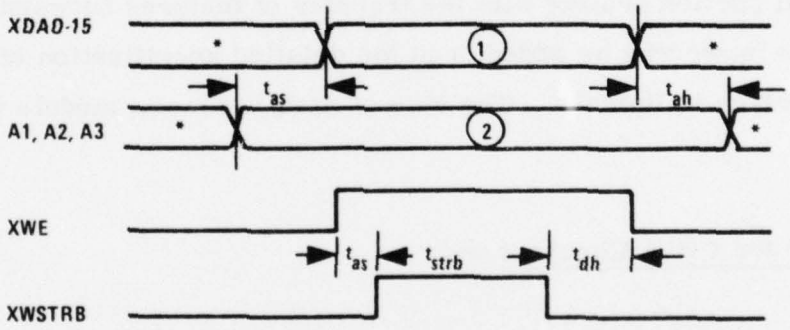
NOTE 2: HIGH WHEN BYTE WRITE INTO MEMORY

NOTE 3: RPLY MUST COME IN 1000ns OR LESS AFTER DIN OR THE DMA-L11 WILL HANG IN T53 UNTIL IT IS RECEIVED. IF RPLY DOES NOT COME IN 10 μs THE BUSERR FLIPFLOP IS SET AND THE DMA TRANSFER IS TERMINATED

Figure 33. DMA-L11 DMA Timing: Read From Memory



t_{as} = ADDRESS SETUP TIME $\geqslant$ 100ns
t_{ds} = DEVICE ACCESS TIME $\leqslant$ 450ns
t_{dh} = DEVICE DATA HOLD TIME $\geqslant$ 250ns
t_{RE} = READ ENABLE TIME $\geqslant$ 700ns
t_{ah} = ADDRESS HOLD TIME $\geqslant$ 100ns
1 = DATA SETUP ON BUS
2 = STABLE DATA ON BUS
3 = STABLE ADDRESS
* = DON'T CARE



t_{as} = ADDRESS SETUP TIME $\geqslant$ 100ns
t_{ds} = DATA SETUP TIME $\approx$ 60ns
t_{strb} = WRITE STROBE TIME $\simeq$ 370ns
t_{dh} = DATA HOLD TIME $\geqslant$ 270ns
t_{ah} = ADDRESS HOLD TIME $\geqslant$ 100ns
1 = STABLE DATA ON BUS
2 = STABLE ADDRESS
* = DON'T CARE

Figure 34. Programmed Data Output From DMA-L11

SECTION III

SOFTWARE STATUS

This section presents information on the status of the microcode software development. Diagnostic software which is used in hardware checkout is being utilized. The algorithm software will be checked out once the CPU1 hardware is working.

CPU1 FIRMWARE STATUS

All operational microcode for CPU1 has been written and assembled, except for a small portion dealing with the transfer of features between CPU1 and CPU2; the latter will be added once the detailed specification of the CPU1/CPU2 interface is finished. The size of each firmware module is shown in Table 9.

Microcode for CPU1 Checkout

Checkout of CPU1 will be performed with the use of diagnostic microcode resident in writable control store (WCS). Several subsystems have been identified, and a routine has been or will be written to check out each of them as independently as possible. These subsystems, the order in which they will be checked out, and the major components of which they are comprised are shown in Table 10. Subsystems marked with an asterisk in Table 10 do not yet have checkout routines. A routine has also been written to integrate the subsystems on the ALU and sequencer boards.

TABLE 9. MICROCODE MODULE SIZES

Module	Number of Microinstructions
System initialization	29
Bin matching	300
Clutter classifier	145
Object size determination + median filter	113
Moment features	185
Recognition (KNN) classifier	180
Subroutines: Division	28
Subroutines: Square root	20
Subroutines: Normalization	38
Interrupt routines: End of line	5
Interrupt routines: End of frame	2
	1045
	+ 25 (approx.) for CPU1/CPU2 interface
	1070

These routines are meant to be used in the single step mode; in other words, the user steps through the microinstructions one at a time and checks for various logic conditions after each step. This low-speed checkout is intended to find wiring errors and to determine that all components are functioning properly. Once the low-speed checkout has been successfully completed, a high-speed checkout will be done to determine the optimal CPU1 clock periods for use in real time.

For a given instruction, CPU1 uses one of two clock periods, either a long or a short one, the period being selected automatically according to the instruction type. A clock period can be optimally determined by looping CPU1 on a microinstruction which selects the clock period in question and exhibits the worst case delay over the entire class of instructions which use that period. The period is then shortened until the instruction no longer reliably executes.

TABLE 10. CPU1 CHECKOUT

Board(s)	Subsystem	Major Component(s)	Order of Checkout
Sequencer	Microprogram sequencer	AM 2910	2
Sequencer	Interrupt servicing/control	74SXXX logic	3
ALU	RALU	4 Am 2903's	6
ALU	Memory 1/2 address generation	4 Am 2901's	4
ALU	Condition code/carry-in/shift linkage multiplexing, status register control	Am 2904	5
ALU	Multiplier	TDC 1010J	7
ALU	Memory handshake/control*	74SXXX logic	8
WCS/ALU/ Sequencer	WCS/CPU1 interface	Interconnecting cables	1
Memory 1/ Memory 2	Memory operation*	Memory chips	9
DMA	DMA controllers*	Am 29XX logic	10

Microcode for DMA Controllers

CPU1 contains two microprogrammable DMA controllers; one controls the transfer of first level features from the analog front end to Memory 1, and the other controls the transfer of data to and from CPU2. Each DMA controller has its own 256-word by 32-bit control store which is microprogrammed to fit the application in question. Neither controller has yet been programmed, but but flow charts are being developed. The controllers are needed only after we have started CPU1 algorithm check out, in other words, after CPU1 sequencer, ALU, and memory checkout is complete.

CPU-Compatible Microinstruction Format

No additional microinstruction bits have been defined since the previous quarterly report; the microinstruction length remains at 70 bits. However, the microinstruction format which has been shown in previous quarterly reports is not the format which is being input to CPU1. The former format, while being convenient as a logical model for programming, is not well suited for 8-bit wide microinstruction PROMs since some of its subfields would span an unnecessarily large number of PROM boundaries. This means that if microinstructions in the original format were loaded into the PROMs and the subfields in question changed, an excess number of PROMs would have to be reblown. To remedy this, a new format has been defined which seeks to minimize the number of PROMs which need changing when the contents of isolated subfields are changed. This new format is merely a rearranging of the bits in the old format. Figure 35 shows both the programming and PROM formats and the mapping from one to the other. Software which takes the bits output by the microassembler in the programming format and scrambles them

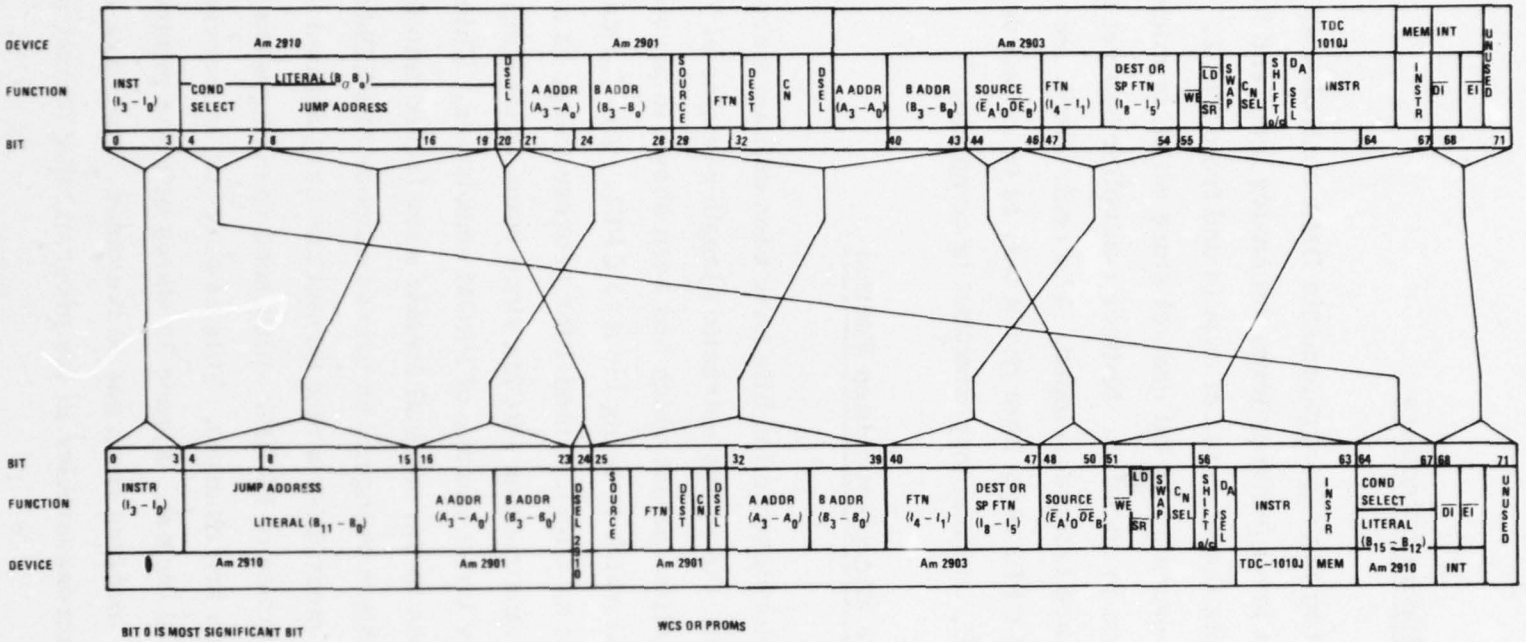


Figure 35. Microinstruction Bit Mapping From Microassembler to CPU1 WCS or PROMs

into the format required by the WCS and PROMs has been developed and is described in the next section.

WRITABLE CONTROL STORE SOFTWARE STATUS

The writable control store (WCS) provides a 72-bit by 2048-word microcode store for CPU1. The memory is interfaced to the Intel MDS-800 microcomputer and is treated as parallel I/O by the MDS system. To facilitate rapid loading and debugging of the program microcode to be resident in the WCS, a number of assembly language routines were developed for the MDS system. These routines, written in 8080 assembly code, are grouped into the two main programs described below.

Scrambling/Loading Program

The first of the two programs is used to generate a "scrambled" microcode file compatible for later use by the Intel PROM programming routines. The "scrambled" microcode file is also loaded into the WCS memory at the addresses determined by the program counter in the microcode file. The term "scrambled" as used above simply refers to the operation of applying a fixed permutation of bits to each microcode word so that the resulting bit pattern is compatible with the pinout of CPU1 and the PROM card. Figure 35 illustrates this bit permutation.

Utility Program

The second 8080 program that was developed consists of a set of utility routines intended for use during the debugging of algorithm microcode. The utility functions provided include:

- Load of previously "scrambled" microcode object file into the WCS memory from diskette.
- Verify the contents of the WCS memory against a specified object file.
- Modify or display selected segments of the WCS memory.
- Perform a test of the memory chips for faults.

Summary

Both programs described above are diskette resident on the Intel MDS system. They each occupy about 1K bytes of RAM to operate and can be executed from the CRT terminal in an interactive fashion.

SOFTWARE STATUS

CPU2 software status has not changed since the last quarterly report.

SECTION IV

PLANS FOR NEXT REPORTING PERIOD

The hardware checkout will continue through the next reporting period. Assuming that the problem of late parts delivery can be overcome, CPU1 checkout should be completed and software microcode checkout will begin. The last design task of interval design should be also completed and all boards should be built.

Preliminary integration of the hardware should be completed and a power supply will be ordered for the chassis. The software for CPU2 will be nearly completed except for certain interfacing details which will have to be cleared up when CPU1 software is checked out.

SECTION V

REVISED SCHEDULE

This section provides revised milestone dates and schedule for the remaining portion of the program. This is based upon anticipated delivery dates for high-speed parts for final checkout and integration.